UMA ABORDAGEM MELHORADA DO ALGORITMO DE OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS PARA O PROBLEMA DE CLUSTERIZAÇÃO DE DADOS

Osires Pires Coelho Filho
Instituto Federal de Educação, Ciência e Tecnologia do Piaui- IFPI
Departamento de Informática
Osires123@gmail.com

Carlos Alberto Martinhon
Universidade Federal Fluminense -UFF
Instituto de Computação
martinhon@ic.uff.br

Lucídio dos Anjos Formiga Cabral
Universidade Federal da Paraíba - UFPB
Departamento de Computação Científica
lucidio@ci.ufpb.br

RESUMO

Clusterização ou Agrupamento de Dados é a técnica mais conhecida em Mineração de Dados. Um problema de Clusterização consiste em agrupar objetos de uma base de dados em grupos de objetos similares de acordo com um conjunto de características. Para resolver este problema tem sido comum o uso da metaheurística PSO - *Particle Swarm Optimization* e suas variações. Neste trabalho propomos uma abordagem que melhora a convergência do PSO através de uma modificação no algoritmo CPSO (*Chaotic* PSO), onde o *K-Means* é usado para acelerar a convergência do algoritmo, melhorando o posicionamento de exatamente uma partícula escolhida aleatoriamente em determinadas iterações do PSO e utilizando uma variável caótica para determinar o índice do vetor do peso inercial w. Esta versão denominada ECPSO (*Enhanced* CPSO) apresentou excelentes resultados quando testada em várias bases de dados de referência da área, obtendo novos melhores valores para algumas delas e igualando as restantes.

Palavras chave: Otimização por Enxame de Partículas, Clusterização, Mapa Caótico

ABSTRACT

Clustering or data grouping is the best known approach in Data Mining. A problem of clustering is to group objects from a database into groups of similar objects according to a set of characteristics. To solve this problem has been usually used the metaheurístic PSO (Particle Swarm Optimization) and its variations. In this paper we propose an approach which improves the convergence of the PSO using a modification of the algorithm CPSO (Chaotic PSO), where the K-means is used to accelerate the convergence of the algorithm, improving the positioning of one particle randomly selected in some iterations of the PSO and using a chaotic variable to determine the index of the array w of inertial weights. This version called ECPSO (Enhanced CPSO) showed excellent results when tested in various databases which are benchmarks in this area and it was able to obtain new best values for some and equating the others.

Keywords: Clustering, Particle Swarm Optimization, Chaotic Maps

1. Introdução

Clusterização de Dados é a técnica mais conhecida em Mineração de Dados. É usada no processo de análise de dados para reconhecimento de padrões. Este processo consiste em reunir em grupos objetos com características similares. Os objetos de um mesmo grupo ou cluster devem ser o mais parecidos entre si e o mais diferente possível de objetos de outros grupos. [Han et al., 2001; Cohen & Castro, 2006]. Este processo de agrupar objetos em grupos exige o processamento de muitas combinações de grupos e objetos até que seja encontrada uma disposição ideal e, portanto, apresenta uma complexidade de ordem exponencial. Isto significa que métodos computacionais que utilizam a "força bruta", como enumerar todos os possíveis grupos e escolher a melhor configuração exigiriam um altíssimo custo computacional e, portanto, não são viáveis. É necessário então buscar uma heurística eficiente que permita resolver o problema num tempo aceitável.

Muitos algoritmos para Clusterização de Dados tem sido propostos, como o K-Means ou k-médias, Algoritmos Genéticos e outros. O K-Means é o algoritmo mais conhecido e vem sendo aplicado com sucesso nos problemas práticos de Clusterização de Dados. Porém, o K-Means pode não ser eficiente na determinação de soluções viáveis de boa qualidade, principalmente quando sua inicialização não for bem sucedida e os centroides iniciais representantes dos clusters ficarem mal posicionados no espaço de busca.

Pesquisas mostram que o comportamento natural dos seres vivos (insetos, abelhas, formigas, cupins, etc.) [Millonas, 1994] é implementado com sucesso para resolver muitos problemas de otimização e portanto nos últimos anos algoritmos baseados nos paradigmas da Computação Evolucionária e Inteligência de Enxames também vem sendo bastante usados para melhorar a Clusterização de Dados. O representante mais popular destas técnicas e que vem obtendo mais êxito é a meta-heurística Otimização por Enxame de Partículas ou Particle Swarm Optimization (PSO) [Merwe and Engelbrecht 2003].

Embora as técnicas baseadas nos paradigmas de Computação Evolucionária e Inteligência de Enxames obtenham na maioria das vezes bons resultados para os problemas de Clusterização, estudos ainda apontam que o uso prático destas técnicas ainda possuem algumas falhas na solução de problemas complexos de otimização, pois são sensivelmente limitadas pelo alto custo computacional e taxa de convergência lenta para uma solução global do problema.

Inúmeras pesquisas [Rana & Kumar, 2011; Kao et al, 2008] demonstraram que o PSO padrão aplicado à Clusterização de Dados possui um desempenho superior aos demais algoritmos usados para o mesmo fim. No entanto, a taxa de convergência na busca do ótimo global ainda é passível de melhora. Várias modificações no PSO básico foram introduzidas para melhorar a velocidade de convergência e a qualidade das soluções encontradas. Essas modificações incluem: a introdução de uma massa de inércia w em relação ao PSO original [Shi & Eberhart, 1998], fixação e constrição da velocidade das partículas [Clerc & Kennedy, 2002], diferentes maneiras de determinar as posições do personal best e global best [Kennedy, 2000] e diferentes modelos de vizinhança [Kennedy, 2000].

Uma das versões principais e mais exitosas variações do PSO é a variação conhecida na literatura como CPSO [Chuang et al., 2011. Nesta variação componentes randômicos são substituídos por sequências de números caóticos ou mapa caótico [Alatas et al., 2009] para multiplicar o componente cognitivo, personal best, e o componente social, global best, respectivamente.

Este trabalho se propõe à criação de uma abordagem melhorada para o uso do algoritmo CPSO na Clusterização de Dados. Nesta versão, além de ser utilizado o número caótico para substituir os componentes randômicos, utilizamos também um número caótico para gerar o índice do vetor que guarda os valores do componente inercial w. Outra inovação desta variação do CPSO é que ele utiliza uma estratégia de aceleração na qual é feita uma chamada ao algoritmo K-Means em determinadas iterações do PSO para melhorar a qualidade de uma partícula aleatoriamente. Isto contribui de forma decisiva para que o algoritmo tenha a velocidade de

convergência melhorada e portanto chegue a uma solução melhor a um custo computacional menor.

Os resultados de testes feitos com a utilização de cinco bases de dados reais com diferentes problemas de Clusterização demonstram que a versão proposta é superior às encontradas na literatura, tais como K-Means, GA, K-GA, PSO, K-PSO, CPSO e ACPSO [Murthy & Chowdhury, 1996; Bandyopadhy & Maulik, 2002; Chuang et al., 2011].

2. Clusterização de Dados

Clusterização de Dados é o processo que visa reunir vetores multidimensionais em grupos, tendo em vista obter a maior similaridade dentro de cada grupo. Esta técnica também conhecida como Agrupamento de Dados é a mais conhecida em Mineração de Dados. Segundo [Jain et al., 1999], Clusterização é a classificação não-supervisionada de dados formando agrupamentos ou clusters. Ela representa uma das principais etapas do processo de análise de dados, denominada análise de clusters. Um Problema de Clusterização consiste em dada uma base de dados X, agrupar (clusterizar) os *n* elementos ou objetos de X de modo que objetos mais similares fiquem no mesmo cluster e objetos menos similares sejam alocados para clusters distintos. Para medir a similaridade entre elementos de um mesmo cluster a métrica mais utilizada é a distância entre eles, quanto mais próximo mais similar e vice-versa. As principais medidas de distância são: Distância Euclidiana, Distância de Minkowsk e Distância de Manhattan (quarteirões). Existem basicamente duas classes de Problemas de Clusterização: o caso onde o número de clusters não é previamente conhecido e o caso mais estudado onde o número de clusters já é previamente conhecido, chamado problema de k-clusterização. Num problema de k-clusterização, o número total de diferentes formas de agrupamento de *n* elementos de um conjunto em k clusters, ou seja, o número de soluções possíveis, cresce exponencialmente e equivale à função N(n, k) apresentada a seguir:

$$N(n,k) = \frac{1}{k!} \sum_{i=1}^{k} (-1)^i \binom{k}{i} (k-i)^n \quad (1)$$

Algoritmos de Clusterização de Dados podem ser classificados em duas categorias principais: agrupamento hierárquico e agrupamento particional. Nos algoritmos da categoria particional, o conjunto de elementos é dividido em *k* subconjuntos, onde cada sub-conjunto é representado por um elemento denominado centróide, e cada configuração obtida é avaliada através de uma função objetivo. As funções objetivo mais utilizadas são a distância intra-cluster: distância de todos os elementos aos seus respectivos centróides, distância inter-cluster: menor distância entre dois centroides e o erro de quantização. Caso a avaliação da função objetivo indique que a configuração não atende ao problema em questão, nova configuração é obtida através da migração de elementos entre os clusters, e o processo continua de forma iterativa até que algum critério de parada seja alcançado. Neste esquema de migração dos elementos entre os clusters, referenciado na literatura como Otimização Iterativa, os clusters podem ser melhorados gradativamente, como é feito no algoritmo de particionamento k-means.

2.1 Algoritmo *K-Means*

Uma das técnicas de particionamento mais conhecidas e utilizadas para Clusterização de Dados é o K-Means [Mitra and Acharya, 2004]. O K-Means usa um elemento para representar cada cluster, chamado de centroide. Este elemento possui um valor médio de todos os atributos dos elementos do cluster. O k-means classifica os elementos em um número de clusters previamente conhecido (k clusters). A ideia principal do algoritmo é estabelecer k centróides iniciais aleatoriamente e em seguida verificar de que centróide o objeto é mais similar, ou seja, está mais próximo, formando clusters ou agrupamentos destes objetos. Este processo é iterativo e a cada iteração os centróides são reposicionados calculando-se a média dos valores dos pontos

mais próximos de cada centroide. Ao reposicionar os centroides os pontos então podem ficar mais próximos de outro centroide na próxima iteração e então farão parte de outro cluster.

Os centróides iniciais são definidos aleatoriamente, o que é a maior desvantagem deste algoritmo, pois quando os centroides são mal posicionados o resultado da Clusterização é ruim.

Algoritmo_*Kmeans*
Início
1 Selecionar os pontos para clusterizar;
2 Inicializar aleatoriamente os centróides;
3 Repetir
4 Atribuir centroide aos pontos de acordo com a similaridade (proximidade)
5 Recalcular/reposicionar centroides;
6 Até que os pontos não mudem de centróides;
Fim_Algoritmo_kmeans;

3. Otimização por Enxame de Partículas (*Particle Swarm Optimization - PSO)*

Muitas espécies de animais se organizam em grupos [Anderson and Franks, 2001] nos quais geralmente há a presença de um líder que controla ou orienta o restante do grupo, como por exemplo o caso de um bando de leões ou uma matilha de lobos, etc. Neste tipo de sociedade organizada, o comportamento dos indivíduos é fortemente ditado pela hierarquia social. Mais curioso ainda é o comportamento das espécies que vivem em grupos em que nenhum líder pode ser identificado, por exemplo, um bando de aves ou um cardume de peixes. Nestes grupos sociais os indivíduos não tem nenhum conhecimento do comportamento global de todo o grupo e nem têm qualquer informação sobre o meio ambiente. Apesar disso, eles tem a capacidade de juntar-se e se moverem juntos, com base em interações locais entre indivíduos, ou seja, um comportamento social complexo surge a partir de interações simples entre indivíduos de uma mesmo grupo ou população.

Com base no comportamento descrito acima conhecido como Swarm Intelligence (SI) [Kennedy et al, 2001; Millonas, 1994], Kennedy e Eberhart desenvolveram o algoritmo conhecido como Particle Swarm Optimization - PSO [Kennedy & Eberhart, 1995]. Este algoritmo segue um método estocástico de otimização. Em analogia com os paradigmas de Computação Evolucionária, um enxame é semelhante a uma população, enquanto que uma partícula é semelhante a um indivíduo. Simplificando, as partículas "voam" através de um espaço de busca multidimensional, em que a posição de cada partícula é ajustada a cada iteração do algoritmo de acordo com a sua própria experiência e a de seus vizinhos. Cada partícula é uma solução viável para o problema e tem sua qualidade ou valor dado por uma função de fitness (aptidão), também chamada de função objetivo, conforme Figura 01 abaixo:

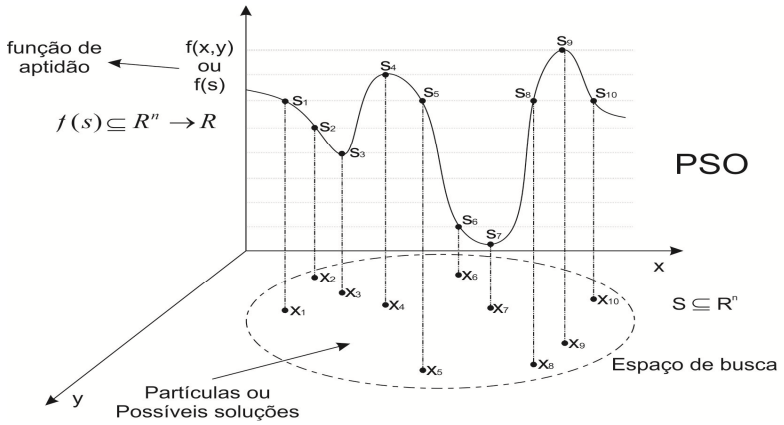


Figura 01 - Partículas = Soluções

Para um enxame de N partículas, a velocidade da i-ésima partícula na posição Xi(t) é calculada pela equação:

$$V_i(t+1) = \underbrace{w_i(t) \cdot V_i(t)}_{\text{Velocidade prévia}} + \overbrace{c_1 \cdot r_1 \cdot \left(Pbest_i(t) - X_i(t)\right)}^{\text{Componente cognitivo}} + \overbrace{c_2 \cdot r_2 \cdot \left(Gbest - X_i(t)\right)}^{\text{Componente social}} \quad (2)$$

Onde c_1 e c_2 são constantes de aceleração, r_1 e r_2 são variáveis randômicas e $Pbest_i(t)$ e Gbest são respectivamente a melhor posição que a partícula i já ocupou e a melhor posição já ocupada por qualquer das partículas até o momento. A variável w, incluída depois na fórmula, é chamada de peso inercial e serve como uma espécie de "freio" que diminui gradativamente a velocidade da partícula a cada iteração. A nova posição da partícula é dada por:

$$\underbrace{X_i(t+1)}_{\text{Posição atual}} = \underbrace{X_i(t)}_{\text{Posição anterior}} + \underbrace{V_i(t+1)}_{\text{Velocidade atual}} \quad (3)$$

Abaixo, na Figura 02, segue a ilustração sobre a movimentação de uma partícula $X_i(t)$ num espaço bidimensional:

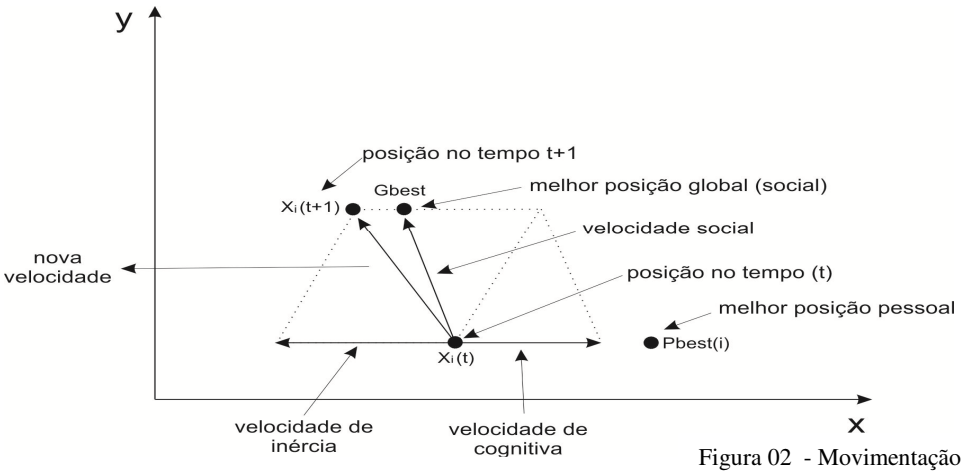


Figura 02 - Movimentação de uma partícula

O vetor velocidade dirige todo o processo de otimização e reflete ambos os conhecimentos da partícula: o conhecimento advindo da sua própria experiência e o adquirido com a troca de informações com sua vizinhança. O conhecimento individual adquirido da sua própria experiência é referenciado como e componente cognitivo da partícula e proporciona a informação sobre a menor distância da solução ideal que aquela partícula já esteve. Esta posição é denominada como posição *personal best (Pbest)* da partícula e influencia somente no seu próprio movimento. A informação obtida da troca de informações com a vizinhança é referenciada como o componente social na equação de velocidade. O componente social é também conhecido como *global best (Gbest)* e influencia no movimento de todas as partículas do enxame. Originalmente o algoritmo PSO tem duas versões que diferem somente em relação à vizinhança das partículas. Os dois algoritmos foram batizados de: PSO *Gbest*, onde a vizinhança de cada partícula é todo o enxame e o PSO *Lbest*, onde uma topologia de vizinhança e seu

tamanho são determinados previamente. Após testes utilizando as duas versões acima, verificamos que a versão PSO Gbest alcança melhores resultados que a versão PSO Lbest em todas as bases de dados testadas, visto que, devido a estratégia utilizada, o ECPSO utilizando o PSO Gbest possui uma maior taxa de convergência e não fica preso em ótimos locais devido a ação do número caótico, conforme será visto na seção 3.2. Na versão do PSO Lbest testada, foi utilizada a topologia de rede em anel tradicional, onde cada partícula tem como vizinhança apenas duas partículas: a imediatamente posterior e a imediatamente anterior em relação ao índice [Engelbrecht, 2005].

3.1 Variações do PSO básico

O PSO básico tem sido aplicado com sucesso a um grande número de problemas de otimização, em especial a problemas de Clusterização [Omran, Engelbrecht, & Salman, 2005] e embora os resultados empíricos apresentados ilustrem a capacidade do PSO para resolver os problemas de otimização, estes resultados também mostram que o PSO básico tem problemas de convergência para boas soluções. Um grande número de modificações no PSO básico foram desenvolvidas para melhorar a velocidade de convergência e a qualidade das soluções encontradas pelo PSO. Essas modificações incluem a introdução de uma massa de inércia, fixação da velocidade, constrição de velocidade, diferentes maneiras de determinar as posições do personal best e global best, e diferentes modelos de velocidade.

3.2 *Chaotic Particle Swarm Optimization - CPSO*

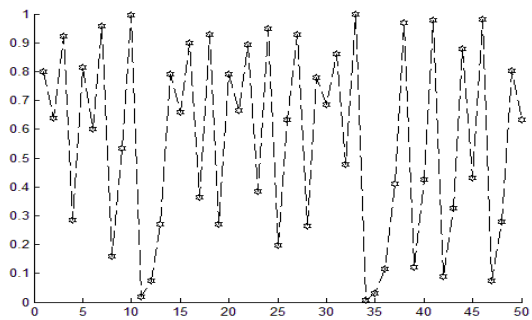
Uma das muitas versões do PSO conhecida como CPSO [Chuang et al., 2011], usa um mapa caótico ou sequência de números caóticos para substituir os componentes randômicos $r_1 e\, r_2$ da Equação 4. No campo da engenharia, é bem reconhecido que a Teoria do Caos pode ser aplicada como uma técnica muito útil em aplicações práticas [Coelho & Mariane, 2009].

Os números caóticos são semelhantes aos números randômicos e são gerados por fórmulas de iteração determinísticas, como o Mapa Logístico, um dos mapas caóticos mais simples, introduzido por [May, 1976], que gera números caóticos usando a seguinte fórmula:

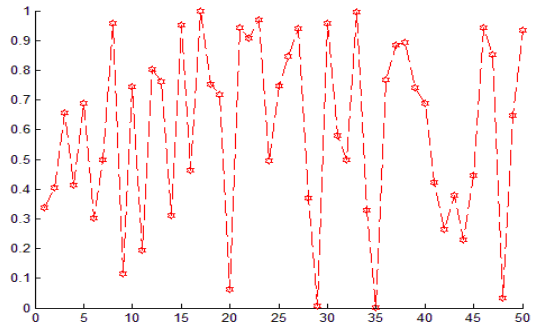
$$\mathrm{Cr}(\mathrm{t}+1) = \mathrm{k} * \mathrm{Cr}(\mathrm{t}) * (1 - \mathrm{Cr}(\mathrm{t})) \qquad (4)$$

Na Equação 4 o Cr(t) inicial é um número randômico entre 0 e 1 e diferente dos valores do conjunto $\{0.00\ 0.25\ 0.50\ 0.75\ 1.00\}$. O valor de k controla o comportamento do Mapa Logístico. Para k = 4 o valor de Cr tem um comportamento caótico.

Os Gráfico 01 abaixo, mostra o comportamento da sequência de números caóticos gerados pelo Mapa Logístico e uma sequência de números gerados aleatoriamente.



Sequência de números caóticos



Sequência de números randômicos

Gráfico 01

Em [Mendel et al., 2011] é feita uma análise estatística que justifica o uso do mapa logístico no PSO. Eles mostram que o mapa logístico com k = 4 gera mais números aleatórios próximos aos dois extremos do intervalo [0,1]. Esta propriedade do mapa logístico pode ser vista no histograma da Figura 03, que mostra o grau de dispersão dos valores gerados por um mapa logístico com k = 4 num intervalo z[0,1]. Desta forma o uso desta característica permite que partículas possam dar "saltos" maiores para escapar de ótimos locais ou sair de uma situação de estagnação mais facilmente, como também dar "saltos" pequenos possibilitando um maior refinamento da pesquisa.

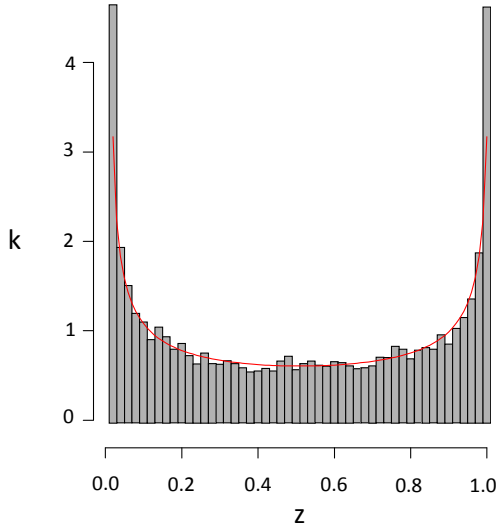


Figura 03 - Histograma de 10.000 iterações do mapa logístico

A equação da velocidade para o CPSO com a inclusão do número caótico Cr gerado pelo Mapa Logístico com valor de k=4 em substituição aos números randômicos fica então da seguinte forma:

$$V_i(t+1) = w_i(t) * V_i(t) + c_1 * Cr(t) * \left(Pbest_i - X_i(t)\right) + c_2 * \left(1 - Cr(t)\right) * (Gbest - X_i(t)) \quad (5)$$

4. *Enhanced Chaotic Particle Swarm Optimization - ECPSO*

No ECPSO utilizado para Clusterização de Dados cada partícula é uma solução em potencial que representa o conjunto dos k centróides necessários para resolver o problema. Cada partícula Xi é construida como: $x_i = (m_{i1}, \ldots, m_{ij}, \ldots, m_{iNc})$, onde m_{ij} refere-se ao j-ésimo vetor de centroides da i-ésima partícula. A dimensão de cada partícula D é dada então pela multiplicação do número de dimensões de cada objeto, *d*, pelo número de centroides ou cluster, *k*, ou seja, $D = k * d$. Abaixo tem-se uma ilustração de partículas representando centróides para Clusterização de objetos de duas dimensões em três clusters.

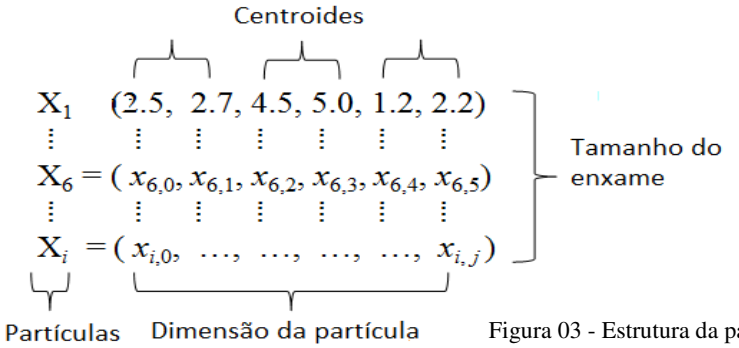


Figura 03 - Estrutura da partícula

Na variação do PSO proposta neste trabalho, procuramos melhorar ainda mais o poder do algoritmo CPSO. Propomos usar um peso inercial *w* com um componente caótico para termos os mesmos benefícios obtidos com o comportamento caótico dos componentes Gbest e Pbest da equação de velocidade original do algoritmo CPSO. Nesta abordagem o índice do vetor *w* é determinado por um número caótico o que faz com que a componente velocidade atual da equação de velocidade tenha um comportamento caótico. A equação de velocidade da versão proposta fica da seguinte forma:

$$V_i(t+1) = w(\mathbf{i\_cr}) * V_i(t) + c_1 * Cr(t) * \left(Pbest_i - X_i(t)\right) + c_2 * \left(1 - Cr(t)\right) * (Gbest - X_i(t)) \quad (6)$$

Onde o *i_cr* é calculado a cada iteração, usando-se uma variável caótica determinada pela Equação 4. Isto permite que a partícula possa fazer movimentos maiores ou menores dentro da faixa pré-definida inicialmente para o vetor de peso inercial *w*, independente da iteração, possibilitando que a partícula fuja de possíveis ótimos locais com mais facilidade.

A estratégia de aceleração utilizada neste algoritmo funciona da seguinte forma: na primeira iteração do algoritmo, o *K-Means* é chamado passando-se a partícula com melhor fitness. Nas iterações que se seguem, até um total de $Max_{iter}/\ 10$, o *K-Means* é chamado mas recebe uma das partículas do PSO escolhida aleatoriamente. Este procedimento faz com que um terço das partículas, que possam ter um bom potencial, sejam melhoradas, possibilitando assim que o algoritmo chegue a uma convergência para uma solução de alta qualidade. O diferencial desta estratégia é que enviamos para o *K-Means* centroides (partículas) melhorados pelo ECPSO, eliminando a desvantagem do K-Means quando recebe centroides mal posicionados. O K-Means devolve então para o ECPSO a partícula melhorada ficando para o ECPSO a tarefa de refiná-la ainda mais. Este procedimento não precisa ser feito em todas as iterações, pois se assim o fosse o *K-Means* seria executado desnecessariamente, pois não conseguiria trazer resultados melhores que os do ECPSO, ou seja, a chamada do *K-Means* só melhora as partículas que recebe do ECPSO quando o ECPSO está nas iterações iniciais.

Abaixo um resumo do procedimento utilizado pelo ECPSO para a Clusterização de Dados.

Algoritmo ECPSO;

Início

1 Inicializar dimensão das partículas;
2 Inicializar posição das partículas X_i;{posições aleatórias no espaço de busca}
3 Inicializar velocidade das partículas em cada dimensão v_{id}
4 Inicializar constantes c_1 *e* c_2 de iterações *iter_max*
5 Inicializar variável de peso inercial *w*
6 Determinar função de aptidão ou função objetivo do problema
7 Determinar valor da aptidão inicial de cada partícula (pbest)
8 Determinar Gbest inicial
9 Enquanto não alcançar número máximo de iterações faça
10 Calcular o índice caótico do vetor do peso inercial ***w***
11 Inicializar aleatoriamente o número caótico *Cr*
12 Para cada partícula faça
13 Calcular para cada dimensão o número caótico *Cr* de acordo com a Eq. 4
14 Atualizar velocidade da partícula em cada dimensão usando Eq. 6
15 Atualizar posição da partícula em cada dimensão usando Eq. 3
16 Calcular nova aptidão da partícula
17 Fim_para
18 Se iteração <= iter_max / 10 então
19 Aplicar estratégia de aceleração (chamada do *K-Means*)
20 Atualizar partícula caso tenha tido melhora no K-Means

21 Fim_se.
22 Atualizar pbest
23 Atualizar gbest
24 Fim Enquanto
Fim_Algoritmo.

5. Resultados Computacionais

Com o objetivo de demonstrar o bom desempenho do ECPSO utilizamos algumas bases de dados de referência para o problema de Clusterização de Dados e comparamos os resultados com sete outros algoritmos da literatura utilizados para resolver o problema. Os programas foram implementados na linguagem de programação MatLab utilizando-se da mesma métrica de desempenho, a soma das distâncias intra-cluster e os parâmetros de configuração descritos a seguir.

5.1 Bases de dados

O experimento consiste em aplicar a versão do ECPSO para a Clusterização de cinco bases de dados de referência: Iris, Vowel, Wine, CMC, e Cancer, que contêm diferentes números de clusters e dimensões, para comparação dos resultados com outros algoritmos de Clusterização, conforme Tabela 01. As bases foram baixadas de ftp://ftp.ics.uci.edu/pub/machine-learning-databases/. A seguir as características das bases utilizadas e suas descrições.

Base de Dados	Clusters	Dimensões	Quantidade de Objetos
Vowel	6	3	871
Iris	3	4	150
CMC	3	9	1473
Cancer	2	9	683
Wine	3	13	178

Tabela 01: Características das bases de dados

1. A base de dados *Vowel* (vogal) é constituído de 871 sons vocálicos indígenas Telugu. Ele inclui as três características correspondentes às primeira, segunda e terceira frequências de vogal, e seis classes que se sobrepõem.
2. A base de dados Iris é composto de três espécies diferentes: Iris setosa, iris virginica e versicolour íris. Para cada espécie, 50 amostras foram coletadas com quatro características cada, comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala.
3. A base de dados Método de Escolha Contraceptiva, conhecida como CMC é composta por amostras que consistem de dados obtidos de mulheres casadas que estavam ou não grávidas ou não tinham certeza do seu estado de gravidez no momento em que as entrevistas foram realizadas. Ele prevê que a escolha do método contraceptivo de uma mulher se dá com base em suas características demográficas e sócio-econômicas.
4. A base de dados *Breast Cancer* (Cancer de mama) de Wisconsin, consiste de 683 objetos identificados por nove características: Espessura, uniformidade de tamanho da célula, forma celular, adesão marginal, tamanho da célula epitelial, núcleos, cromatina, nucléolos normais e mitose.
5. A base de dados *Wine* (vinho) consiste em 178 objetos identificados por 13 características: teor de álcool, quantidade de ácido málico, o teor de cinzas, a alcalinidade das cinzas, a

concentração de magnésio, fenóis, flavonóides, fenóis não-flavonóides e proantocianinas e intensidade de cor, matiz e OD280/OD315 de vinhos diluídos. Estas características foram obtidas por análise química dos vinhos que são produzidos na mesma região, na Itália, mas são derivados de três cultivos diferentes.

5.2 Métodos e parâmetros de configuração

A quantidade de partículas que são usadas no experimento depende de cada base de dados a ser clusterizada. Depende das características dos objetos e do número de clusters. São inicializadas aleatoriamente *3*N* partículas, onde *N* é a dimensão de cada partícula e é dado por: $N = k * d$. Onde *k* representa o número de cluster a ser informado previamente e *d* representa o número de dimensões ou características de cada objeto. Cada partícula que representa os *k* centroides da solução recebe valores aleatórios entre os máximos e mínimos valores de cada dimensão dos objetos a serem clusterizados. Por exemplo: dados objetos com duas características ou dimensões para serem clusterizados em três clusters, então teremos a dimensão da partícula $N = k * d => N = 3 * 2 = 6$.

O número de máximo de iterações realizadas pelo ECPSO também é dependente dos objetos e do número de cluster e é dado por $\text{Max}_{\text{iter}} = 10 \text{ x N}$. No caso do exemplo acima teríamos o número máximo de iterações igual a $\text{Max}_{\text{iter}} = 10 \text{ x } 6 => Max_{\text{iter}} = 60$.

Os coeficientes de aceleração, constantes $c_1 \, e \, c_2$ combinados com as variáveis randômicas $r_1 \, e \, r_2$, controlam a influência estocástica dos componentes social e cognitivo na velocidade da partícula. Os valores de $c_1 \, e \, c_2$ são definidos como 1.5 e 2.0 respectivamente. O valor de c_1 maior que c_2 faz com que as partículas tenham a tendência a se aproximarem um pouco mais rapidamente em direção a um ótimo global.

O peso inercial ***w*** é inicializado com valores entre ***0.6 e 0.2***. A cada iteração um valor de ***w*** nesta faixa é escolhido de acordo com o valor do número caótico *Crw* (Eq. 4) calculado para cada iteração, desta forma ***w*** pode assumir valores grandes ou pequenos, possibilitando assim que a partícula possa dar "passos" maiores e possa fugir de um ótimo local mesmo no final das iterações.

O valor de fitness utilizado é a soma das distâncias intra-cluster. Quando a soma das distâncias é pequena, o resultado da clusterização é considerado excelente. O valor de aptidão de cada partícula pode ser calculado com a seguinte função:

$$\text{fitness} = \sum_{i=1}^{k} \sum_{j=1}^{n} (X_i - Z_j) \quad (7)$$

Na Eq. 7 acima, *k* e *n* são os números de clusters e de objetos do conjunto de dados respectivamente. Z_i é o valor do centroide i e X_j é o valor do objeto de dados *j*.

A medida utilizada para determinar a pertinência de um objeto a um cluster foi a menor distância euclidiana.

5.3 Resultados experimentais e discussões

O algoritmo proposto foi implementado utilizando a linguagem de programação MatLab versão 7.11.0 (R2010B). De modo a demonstrar a eficiência do ECPSO, foram comparados os resultados obtidos com os seguintes algoritmos: *K-Means*, GA, KGA, PSO, K-PSO, ACPSO e CPSO. O ECPSO foi executado um total de 20 vezes para cada base de dados utilizada. As medidas reportadas são média aritmética, desvio padrão e melhor resultado obtido nas 20 execuções seguidas.

Tabela 02: Resultados experimentais

Base		*K-Means*	GA	KGA	PSO	K-PSO	CPSO	ACPSO	ECPSO
Vowel	Média	159242.87	390088.24	149368.45	168477.00	149375.70	151337.00	149051.84	**149020.11**
	Desvio	916	N/A	N/A	3715.73	155.56	3491.43	67.27	**47.65**
	Melhor	149422.26	383484.15	149356.01	163882.00	149206.10	148996.50	148970.84	**148967.27**
Iris	Média	106.05	135.40	97.10	103.51	96.76	96.90	96.66	**96.65**
	Desvio	14.11	N/A	N/A	9.69	0.07	0.303	0.001	**0.00**
	Melhor	97.33	124.13	97.10	96.66	96.66	96.66	96.66	**96.65**
CMC	Média	5693.60	N/A	N/A	5734.20	5532.90	5532.23	5532.20	**5532.18**
	Desvio	473.14	N/A	N/A	289.00	0.09	0.04	0.01	**0.00**
	Melhor	5542.20	N/A	N/A	5538.50	5532.88	5532.19	5532.19	**5532.18**
Cancer	Média	2988.30	N/A	N/A	3334.60	2965.80	2964.49	2964.42	**2964.38**
	Desvio	0.46	N/A	N/A	357.66	1.63	0.12	0.03	**0.00**
	Melhor	2987	N/A	N/A	2976.30	2964.50	2964.40	2964.39	**2964.38**
Wine	Média	18061.00	N/A	N/A	16311.00	16294.00	16292.90	16292.31	**16292.18**
	Desvio	793.21	N/A	N/A	22.98	1.70	0.78	0.03	**0.00**
	Melhor	16555.68	N/A	N/A	16294.00	16292.00	16292.19	16292.18	**16292.18**

Os resultados de GA podem ser encontrados em [Murthy & Chowdhury, 1996], os resultados de KGA podem ser encontrados em [Bandyopadhy & Maulik, 2002]. Os resultados de K-Means, PSO, K-PSO, CPSO e ACPSO podem ser encontrados em [Chuang et al., 2011].. N/A: sem dados avaliados.

Verifica-se com base na tabela acima que o ECPSO obteve melhores resultados tanto na média aritmética, desvio padrão e no melhor resultado em todas as bases de dados testadas. Isso se deve pelo fato de que na versão aqui proposta uma partícula de cada iteração é melhorada com o K-Means. Esta partícula é escolhida aleatoriamente mas já tem um bom potencial, pois ela já está melhorada pelo próprio PSO. Diferentemente da versão denominada de ACPSO que melhora as partículas como estratégia de aceleração, mas escolhe um terço das partículas antes que elas entrem no PSO e portanto estas partículas ainda podem estar mal posicionadas enviando centroides ruins para o K-Means. Outro motivo da excelente performance do algoritmo proposto é a facilidade de fugir de ótimos locais com a utilização de um valor do peso de inércia caótico, mas dentro da faixa especificada no inicio do algoritmo.

Para cada base utilizada tem-se um valor de desvio-padrão obtido a partir do respectivo conjunto de soluções geradas pelas execuções. Observamos que os valores alcançados para o desvio padrão são nulos, exceto para a base Vowel, onde alcança um valor extremamente baixo que representa um coeficiente de variação de 0,031975 (desvio-padrão/média). Ou seja, basicamente não há variabilidade, o que denota a robustez do algoritmo ECPSO e que evita a necessidade de usarmos um tratamento estatístico mais elaborado.

6. Conclusões

Neste trabalho foi apresentado uma versão melhorada da hibridização feita do algoritmo PSO e *K-Means* para resolver o problema da Clusterização de Dados, o ECPSO. Nesta versão foram introduzidas duas modificações importantes em relação às hibridizações anteriores: utilizou-se um número caótico para gerar o valor do índice do peso inercial e foi inserido ainda uma estratégia de aceleração que possibilitou uma convergência mais rápida do algoritmo. Os resultados foram testados utilizando-se cinco bases de dados de referência e comparados com o desempenho de sete algoritmos utilizando a mesma métrica, ou seja, minimização da distância intra-cluster, para pesquisar centroides ideais num espaço euclidiano N-dimensional. Verificou-se que o ECPSO chega a resultados melhores e com um menor custo computacional que outros algoritmos comparados. Pretende-se ainda, como trabalhos futuros, realizar a hibridização do PSO com outra meta-heurística como VNS, ILS ou GRASP, para superar as desvantagens de algoritmos de busca local como o *K-Means*.

Referências

Alatas, B., Akin, E., & Ozer, A. B. (2009). *Chaos embedded particle swarm optimization algorithms*. Chaos, Solitons & Fractals, 40, 1715–1734.

Anderson C. and N. R. Franks. *Teams in Animal Societies*. *Behavioral Ecology*, 12(5):534-540, 2001.

Chuang, Li-Yeh ; Hsiao, Chih-Jen ; Yang, Cheng-Hong**.** *Chaotic particle swarm optimization for data clustering.* Expert Systems With Applications, 2011, Vol.38(12), pp.14555-14563.

Coelho, L. d. S., & Mariani, V. C. (2009). *A novel chaotic Particle Swarm Optimization approach using Hénon map and implicit filtering local search for economic load dispatch*. Chaos, Solitons & Fractals, 39, 510–518.

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.

Engelbrecht, A. P., Fundamentals of Computational Swarm Intelligence. (John Wiley, 2005).

Han, J., Kamber, M., & Tung, A. K. H. (2001). *Spatial clustering methods in data mining:A survey.* London: Taylor & Francis.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). *Data clustering: A review*. ACM Computing Surveys, 31, 264–323.

Kao, Y.-T., Zahara, E., & Kao, I. W. (2008). A hybridized approach to data clustering. Expert Systems with Applications, 34, 1754–1762.

Kennedy, J., R.C. Eberhart, and Y. Shi. Swarm Intelligence. Morgan Kaufmann, 2001.

Kennedy, J. Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance. In Proceedings of the IEEE Congress on Evolutionary Computation, Volume 2, pages 1507-1512, July 2000.

Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. In IEEE international joint c conference on neural network (Vol. 4, pp. 1942–1948).

M. Clerc and J. Kennedy. The Particle Swarm-Explosion, Stability and Convergence in a Multidimensional Complex Space. IEEE Transactions on Evolutionary Computation, 6(1):58-73, 2002.

May, R. M. (1976). Simple mathematical models with very complicated dynamics. Nature, 261, 459–467.

Mendel, E. ; Krohling, R. A. ; Campos, M.. Swarm Algorithms with Chaotic Jumps Applied to Noisy Optimization Problems. Information Sciences, v. 181, p. 4494-4514, 2011.

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.*

Millonas, M. M. Swarms, phase transitions, and collective intelligence. In C.G. Langton (Ed.), Artificial Life III, pp. 417–445. Reading, MA: Addison-Wesley. 1994.

Mitra S. and Acharya T., 2004. Data Mining. Wiley Publications.

Murthy, C. A., & Chowdhury, N. (1996). In search of optimal clusters using genetic algorithms. Pattern Recognition Letters, 17, 825–832.

Omran, M., Engelbrecht, A. P., & Salman, A. (2005). Particle Swarm Optimization method for image clustering. International Journal on Pattern Recognition and Artificial Intelligence, 19, 297–322.

Rana, S., Jasola, S., Kumar, R. A review on particle swarm optimization algorithms and their applications to data clustering. Artificial Intelligence Review 35, pp 211–222, 2011.

Shi, Y., and Eberhart, R. C. (1998). A modified particle swarm optimizer. In The 1998 IEEE international conference on evolutionary computation proceedings (pp. 69–73).