Revista digital
Matemática, Educación e Internet
(http://tecdigital.tec.ac.cr/revistamatematica/).
Vol 19, No 1. Agosto – Marzo, 2019 ISSN 1659 -0643

El algoritmo PSO aplicado al problema de particionamientode datos cuantitativos

Jeffry Chavarría M.
jchavarria@itcr. ac. cr
Escuela de Matemática
Instituto Tecnológico de Costa Rica

Juan José Fallas M.
jfallas@itcr. ac. cr
Escuela de Matemática
Instituto Tecnológico de Costa Rica

Recibido: Mayo 10, 2017 Aceptado: Marzo 20, 2018

Resumen. Este artículo muestra la aplicación del algoritmo de Enjambre de Partículas (Particle Swarm Optimization: PSO) al problema de particionamiento de datos cuantitativos. Se enfatiza particularmenteen la calibración de los parámetros de este algoritmo y el rendimiento que éste muestra en función de los valores asignados a los parámetros. Finalmente, se muestra la aplicación del algoritmo a diez tablas de datos cuantitativos. En este experimentose obtuvieron buenos resultados en términos de la convergencia del algoritmo y los tiempos de ejecución.

Palabras clave: Heurísticas, optimización, enjambre, partículas, particionamiento de datos.

Abstract. This paper shows how the Particle Swarm Optimization algorithm (PSO) was applied to the partitioning of quantitative data. The parameter fitting process for this algorithm is particularly emphasized. Also, this paper illustrateshow the values assigned to the parameters affect the performance of the algorithm. Finally, the application of the algorithm to ten quantitative data tables is shown. In this experiment good results were obtained in terms of the algorithm convergence and running times.

KeyWords: Heuristics, optimization, PSO, swarm, particles, data clustering.

1.1 Introducción

En el problema de particionamiento de datos cuantitativos se considera el conjunto $X = \{x_1, \ldots, x_n\}$, con $x_i \in R^p$ y se busca construir una partición $P = \{C_1, \ldots, C_K\}$ de X en K clases, K dado a priori. Se tiene que $C_l \subset X$; $C_l \neq \emptyset$ y para cada $l, l' = 1, \ldots, n$ y $l \neq l'$ se satisface que $C_l \cap C_{l'} = \emptyset$ y $\bigcup_{l=1}^{K} C_l = X$.

Para la construcción adecuada de P es necesario emplearuna función de distancia que permita cuantificar qué tan similares o disímiles son dos elementos cualesquiera de X con respecto a algún atributo de

interés. Si bien es posible utilizar cualquier función de distancia, para esteestudio se utiliza la distancia euclídea clásica definida como $d : R^p \times R^p \to R$ tal que

$$d^2\left(x_i, x_j\right) = \left(x_i - x_j\right)^T \left(x_i - x_j\right) = \left\|x_i - x_j\right\|^2$$

Además, se considera el centro de gravedad de la clase C_l, como el elemento de R^p denotado por g_l y definido por

$$g_l = \frac{1}{Card\left(C_l\right)} \sum_{x_i \in C_l} x_i,$$

el cual se puede interpretar como un individuo promedio de los objetos que pertenecen a C_l. Lo anterior permite definir la función de inercia intraclases de P como:

$$W\left(P\right) = \sum_{l=1}^{K} \sum_{x_i \in C_l} \left\|x_i - g_l\right\|^2.$$

Al minimizar $W(P)$ se determina la solución al problema de particionamiento. Sin embargo, se tiene el inconveniente que esta función no es convexa, lo cual podría implicar la existencia de muchos mínimos locales (Ng & Wong, [11, (2002)]; Sarkar & Yegnarayana, [13, (2002)]). Esta característica ocasiona que algoritmos tradicionales usados en particionamiento de datos, tales como el algoritmo de $k-$ medias, encuentren mayoritariamente mínimos locales (Trejos & Murillo, [16, 2004]). De manera similar, Babu & Murty [2, 1994] indican que la mayoría de los métodos tradicionales usados en particionamiento de datos corresponden a técnicas de descenso de gradiente que, por su naturaleza de optimalidad local, convergen a soluciones suboptimales de la función objetivo. Estas características representan un fundamento para la búsqueda de estrategias alternativasde optimización, entre las cuales se encuentran las heurísticas de optimización combinatoria.

El algoritmo de Enjambre de Partículas (PSO, por sus siglas en inglés) es una técnica computacional evolutiva inspirada en el comportamiento social de los enjambres, tales como las parvadas de aves, los enjambres de insectos y los cardúmenes de peces, durante el proceso de exploración en búsqueda de alimento y refugio (Chi-Yang & I-Wei, [5, 2011]). Este algoritmo fue propuesto en 1995 por Eberhart & Kennedy [6, 1995], durante una investigación en la que trataban de simular los movimientos sincronizados de bandadas de pájaros, como parte de un estudio socio-cognitivo que investigaba la noción de la "inteligencia colectiva" (Eberhart & Kennedy, [6, 1995]).

La optimización con PSO consiste en un enjambre artificial de individuos, los cuales son llamados *partículas*, en el que cada una de las partículas debe moverse por el espacio multidimensional en búsqueda de soluciones suboptimales (Lima & Barán, [10, 2006]; Sedighizadeh & Masehian, [14, 2009]). El algoritmo se basa en tres principios fundamentales para realizar los movimientos: *evaluación* (capacidad de autoevaluarse) , *comparación* (compararse con partículas vecinas) e *imitación* (seguir los movimientos de las mejores partículas). Este colectivo es lo que permite una evolución en el tiempo con el objetivo de encontrar grupalmente buenas soluciones al problema de optimización.

Este algoritmo ha sido ampliamente utilizado con buenos resultados en el problema de particionamiento. En Kao, Tsai & Wang [8, 2007] se propone un algoritmo de optimización basado en PSO que se aplica al particionamiento de datos para el procesamiento de imágenes. Por su parte, Chi-Yang & I-Wei [5, 2011] también aplicaron esta heurística al particionamiento de datos, pero haciendo énfasis en un ajuste particular deparámetros que acelera la convergencia del algoritmo (dando mayor peso al

parámetro social) y, además, en un mecanismo de regeneración selectiva de partículas que permite el escape de óptimos locales.

En el mismo contexto, Tsai et al. [17, 2015] proponen un algoritmode alto rendimiento basado en enjambres de partículas para la reducción de la complejidad del tiempo de ejecución. El método propone el uso de dos operadores: un operador de reducción (compresión de patrones poco probables) y un operadorMultiStart (diversidad para evadir óptimos locales).

Finalmente, Tan [15, 2015] propone una mejora del método de $k-$ medias basada en PSO. El algoritmo realiza una selección de centros de agrupación iniciales que permite ajustar dinámicamente el coeficiente de inercia de cada partícula, aprovechando la aptitud del grupo para decidir el tiempo de conversión entre el algoritmo PSO y $k-$ medias.

El presente artículo muestra la aplicación del algoritmo PSO al particionamiento de datos cuantitativos, similar a Chi-Yang & I-Wei [5, 2011] , pero haciendo un énfasis mayor al proceso de calibración de parámetros, y sin dar a priori un peso a ninguno de ellos, como lo hacen estos autores. El aporte principal del artículo es el experimento que se realiza para la selección de los parámetros que optimizan el rendimiento del algoritmo. El análisis emplea mallas bidimensionales que permiten observar la forma en la que el algoritmo converge hacia el valor de referencia de la inercia intraclases, que se tiene registrado para ciertas tablas de datos cuantitativos. Este análisis deja ver la forma en la que los parámetros del algoritmo PSO interactúan entre sí, e inciden en la convergencia del algoritmo y en los tiempos de ejecución. Adicionalmente, se utilizan diez tablas de datos que son comúnmente empleadas para realizar pruebas de ejecución en algoritmos de particionamiento de datos cuantitativos. Con ello se busca cuantificar la respuesta del algoritmo PSO, midiendo su velocidad de convergencia y el porcentaje de atracción, que representa una proporción del número de veces que el algoritmo atina la inercia intraclases de referencia en cada tabla, en un cierto número de ejecucionesrealizadas en condiciones similares. El documento se organiza de la siguiente manera. En la sección 2 se presenta la formulación del algoritmo de PSO. La sección 3 explica los experimentos realizados; en particular el proceso seguido para la calibración de los parámetros del algoritmo. En la sección 4 se describen las tablas de datos utilizadas y se muestran los resultados de la aplicación del algoritmo. Finalmente, la sección 5 expone las conclusiones.

1.2 Formulación del algoritmo

Se supone que se quiere minimizar $f : \Omega \subset R^N \to R$, con $N = K \cdot p$, donde Ω es el conjunto de soluciones factibles y cada elemento en Ω es una partícula de la forma $P = (g_1, g_2, \ldots, g_K)$, donde g_l es el centro de gravedad de la clase l, para $l = 1, \ldots, K$. El algoritmo PSO considera un vector $\xi = (P_1, \ldots, P_m)$, que se le denomina*enjambre*, y es tal que $P_k \in \Omega$, para $k = 1, \ldots, m$ (m es el número de partículas en el enjambre).

Las partículas, en un principio estáticas, deberán ser impulsadas en cada iteración. La velocidad y la dirección del movimiento deben incluir los principios de evaluación, comparación e imitación. Para esto es necesario que cada partícula P_k :

1. Conozca su posición actual, que se denota $\overrightarrow{\chi_k} = (\chi_{k1}, \chi_{k2}, \ldots, \chi_{kN})$.
2. conozca la velocidad $\overrightarrow{v_k}$ con la quellegó a $\overrightarrow{\chi_k}$, donde $\overrightarrow{v_k} = (v_{k1}, v_{k2}, \ldots, v_{kN})$

3. Recuerde la mejor posición en la que ha estado en su recorrido, denotada

$$\overrightarrow{\chi p_k} = (p_{k1} p_{k2}, \ldots, p_{kN}).$$

4. Reconozca la mejor posición global del enjambre, encontrada por cualquiera de las partículas. Esa posición se denota

$$\overrightarrow{\chi m} = (m_1, m_2, \ldots, m_N).$$

Para actualizar la posición $\overrightarrow{\chi_k}^{(t)}$ de la partícula P_k en la iteración t, se sigue el siguiente proceso que puede ser consultado con más detalle en Bratton & Kennedy [3, (2007)] , Kennedy & Eberhart [9, (2001)], o Xie et al. [19, (2002a)]. La nueva posición de P_k se actualiza con

$$\overrightarrow{\chi_k}^{(t+1)} \leftarrow \overrightarrow{\chi_k}^{(t)} + \overrightarrow{v_k}^{(t+1)},$$

donde la $j-$ ésima componente del vector $\overrightarrow{v_k}^{(t+1)}$, $j = 1, \ldots, N$ está dada por

$$\overrightarrow{v_{kj}}^{(t+1)} = \alpha \overrightarrow{v_{kj}}^{(t)} + \kappa_1 \left[p_{kj}^{(t)} - x_{kj}^{(t)} \right] + \kappa_2 \left[m_j^{(t)} - x_{kj}^{(t)} \right],$$

con $\kappa_1 = \text{rnd}\,]0, c_1[$ y $\kappa_2 = \text{rnd}\,]0, c_2[$ (la notación "rnd" implica seleccionar un número aleatorio en el intervalo indicado).
Además, α, c_1 y c_2 son parámetros que se deben ajustar y que tienen las siguientes funciones:

- α : parámetro inercial que regula la influencia que posee la velocidad $\overrightarrow{v_k}$, de la iteración t,en el cálculo de la velocidad en la iteración $t+1$.
- c_1 : parámetro cognitivo que controla la influencia que tiene la mejor solución que ha encontradola partícula P_k en su recorrido, sobre el cálculo de la velocidad en la iteración $t+1$.
- c_2 : parámetro social que representa el peso que tiene la mejor solución encontrada por el enjambre, sobre el cálculo de $\overrightarrow{v_k}^{(t+1)}$.

La Figura 1.1 resume cómo P_k (resaltada con color azul) recalcula su nueva posición tomando en cuenta los siguientes componentes: la mejor que ha tenido en su recorrido (posición en verde), la posición que tiene el líder o mejor partícula (partícula en rojo) y, finalmente, un efecto inercial (la partícula tiende a continuar en la dirección que trae en sudesplazamiento). Estas tres componentes vectoriales se combinan para generarel nuevo movimiento que seguirá la partícula en la siguiente iteración.

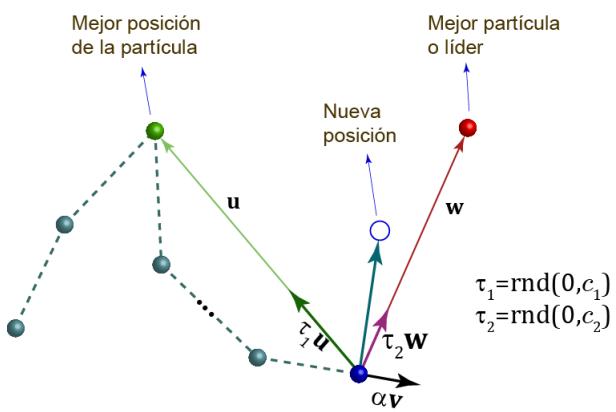

Figura 1.1: Movimiento de una partícula.

Como método de parada, varios autores sugieren detener el algoritmo cuando se hayan ejecutado un número máximo de iteraciones (Alias et al., [1, (2011)]; Lima & Baran, [10, (2006)]; Xie et al., [19, (2002a)]). Por su parte, Hassan et al. [7, (2005)] propone considerar un factor de estancamiento. Esto es, detenerlo si ha transcurrido un cierto número de iteraciones en el que éste no reporta ninguna mejora a la mejor solución encontrada por el enjambre.

El Algoritmo 1 corresponde al algoritmo PSO, empleando las notaciones previamente definidas. El método de $k-$medias es aplicado como estrategia complementaria para mejorar el rendimiento. Esto consiste en que si el enjambre no encuentra una mejor solución, comparada con la que se tiene almacenada, durante un cierto número de iteraciones (parámetro de entrada KM), entonces se hace tender cada partículadel enjambre a un óptimo local, mediante la aplicación del método de $k-$medias clásico. En el Algoritmo 2 se puede visualizar esta estrategia. En este algoritmo se utiliza la variable $\varepsilon = 0.01$ como una tolerancia para las iteraciones en la aplicación de $k-$medias. Esto es, a la partícula P_i se le aplica $k-$medias hasta que la diferencia entre su inercia intraclases de una iteración a la siguiente(la inercia se recalcula de una iteración a otra como consecuencia del cambio en los centroides) sea menor que ε. El método se aplica con dicha tolerancia ($\varepsilon = 0.01$) para tener mayor eficiencia en el tiempo de ejecución del algoritmo.

Como método de parada se aplica el propuesto por Hassan et al. [7, (2005)] , analizando un factor de estancamiento (el parámetro se denota `MaxIteraSinMejora`). En este sentido, en el Algoritmo 1 se usa un contador para las iteraciones en las que el algoritmo no reporta ninguna mejora `ContadorIteraSinMejora`, con respecto a la mejor solución previamente encontrada por el enjambre. Así, la condición de parada utilizada en el MIENTRAS (ver línea 3 del Algoritmo 1) es

`ContadorIteraSinMejora <MaxIteraSinMejora`

Por otra parte, posterior a la actualización del contador de iteraciones globales en la línea 11 del Algoritmo 1, a todas las partículas del enjambre se les aplica condicionadamente el método de $k-$medias expuesto en el Algoritmo 2. Esto es, se aplica el Algoritmo 2 siempre que `ContadorIteraSinMejora` sea un múltiplo de KM. Posterior a este proceso termina el bucle MIENTRAS, regresando nuevamente a la línea 3 del Algoritmo 1.

ALGORITMO 1 Algoritmo PSO

Entrada: Parámetros K, M, α, c_1, c_2 y m.

1: Construya aleatoriamente el enjambre $\xi = (P_1, \ldots, P_m)$.

2: Inicie en 0 `aContador` y `ContadorIteraSinMejora`

3: **MIENTRAS** No se dé el criterio de parada **HACER**

4: Evalúe el costo de cada partícula: $W(P_k)$

5: **PARA** $k \leftarrow 1$ **HASTA** m **HACER**

6: Actualice la velocidad $\overrightarrow{v_k}^{(t+1)}$ de P_k como

$$\overrightarrow{v_{kj}}^{(t+1)} \leftarrow \alpha \overrightarrow{v_k}^{(t)} + \kappa_1 \left[p_{kj}^{(t)} - x_{kj}^{(t)} \right] + \kappa_2 \left[m_j^{(t)} - x_{kj}^{(t)} \right]$$

7: Mueva a P_k a su nueva posición mediante:

$$\overrightarrow{\chi_k}^{(t+1)} \leftarrow \overrightarrow{\chi_k}^{(t)} + \overrightarrow{v_k}^{(t+1)}$$

8: Actualice la mejor posición encontrada por P_k, encaso que sea mejor que la existente. Luego, actualice $\overrightarrow{\chi p_k}$ y reinicie la variable `ContadorIteraSinMejora`.

9: Actualice la mejor posición encontrada por el enjambre, en caso que sea mejor que la existente. En ese caso,
actualice $\overrightarrow{\chi m}$

10: **FIN PARA**

11: Incremente a `Contador`

12: Aplique $k-$medias a P_k si `ContadorIteraSinMejora` es un últiplo de KM.

13: **FIN MIENTRAS**

14: **Retornar** La mejor solución encontrada.

ALGORITMO 2 Método de $k-$medias.

Entrada: Partícula P_i.

1: $\text{Inercia}_{\text{Anterior}} \leftarrow -1$.

2: **MIENTRAS** $|\text{Inercia}_{\text{Anterior}} - W(P_i)| > \varepsilon$ **HACER**

3: $\text{Inercia}_{\text{Anterior}} \leftarrow W(P_i)$.

4: Determine la clasificación, dados los centroides en el vector de coordenadas $\overrightarrow{\chi_i}$.

6: Determine los centroides de cada una de las clases, dada la clasificación calculada en el paso anterior.

7: Actualice $W(P_i)$

8: **FIN MIENTRAS**

9: **Retornar** Partícula P_i modificada.

1.3 Experimentación

Se realizó una fase preliminar de análisis de parámetros con el objetivo de determinar valores adecuados para α, c_1 y c_2. En ese sentido, Lima & Baran [10, (2006)] , así como Sedighizadeh & Masehian [14, (2009)] , indican que típicamente en sus investigaciones asignan $\alpha = 0{,}8$ y $c_1 = c_2 = 2$. Por su parte, Hassan et al. [7, (2005)] proponen rangos para dichos parámetros, de tal manera que $\alpha \in [0{,}4; 1{,}4]$, $c_1 \in [1{,}5; 2]$ y $c_2 \in [2; 2{,}4]$, recomendando finalmente la combinación $\alpha = 0{,}5$ y $c_1 = c_2 = 1{,}5$. Dado que no existe un criterio unificado sobre los valores que deben seleccionarse, se analizó el rendimiento del algoritmo como función de diferentes combinaciones para los parámetros.

Para este análisis se consideró la variable $w = c_1 + c_2$, que relaciona los parámetros cognitivo y social. Además, se extendieron los rangos expuestos en la literatura, para realizar una exploración más extensa. En efecto, se analizó $\alpha \in [-2, 2]$, $c_1 \in [0; 4{,}5]$ y $w \in [0{,}5; 5]$.

Para soporte del proceso se utilizaron dos tablas de datos generadas experimentalmente en Pacheco et al. [12, (2006)], siguiendo una distribución normal de números pseudoaleatorios. Estas tablas se denominarán T105 ($n = 105$, que es la cantidad de objetos por clasificar) y T525 $(n = 525)$.

El agrupamiento fuedesarrollado, en cada tabla, considerando $K = 7$ (número de clases), tal que seis clases tienen varianza $\sigma^2 = 1$ y la clase restante tiene varianza $\sigma^2 = 3$. Además, T105 fue construida con una clase "grande" de cardinalidad 51 y las seis clases restantes con cardinalidad 9. De manera similar, T525 tiene una clase de tamaño 261 y las seis restantes de 44 objetos. Dado que el diseño es controlado, Pacheco et al. (2006) pudieron determinar a priori el valor de $W(P)$ (inercia intraclases) que representa, en cada caso, el agrupamiento de referencia en 7 clases. La Tabla 1.1 muestra los valores de referencia de $W(P)$, para cada tabla.

Tabla	W(P) de referencia
T105	7,6247
T525	7,4561

Tabla 1.1: Valores de referencia de $W(P)$

Para el proceso de ajuste de parámetros se utilizó $m = 10$, `MaxIteraSinMejora=10` y $KM = 3$. Se cuenta con los resultados obtenidos con la tabla T105, registrados mediante mallas bidimensionales. Se construyó una malla para cada valor de α, el cual fue variado desde -2 hasta 2, a paso de 0,25. Para sintetizar la información presentada en este artículo, en la), se muestran únicamente cuatro de las mallas construidas para el análisis de T105. La intensidad de los colores en cada figura es un indicador del rendimiento. Por ejemplo, las zonas resaltadas con rojo oscuro del rango de 0,9 a 1, indican las combinaciones en las que se acertó el valor $W(P)$ de referencia en más de un 90 % de 500 corridas múltiples.

Analizando la Figura 2 se observa que los porcentajes de atracción más altos se generan en una franja alrededor de la recta que une los puntos $(0{,}5;0)$ y $(5;4{,}5)$ de ecuación $c_1 = w - 0{,}5$; de donde se tiene la relación $w = c_1 + 0{,}5$, y así $c_2 = 0{,}5$. Este comportamiento también se presentó en todas las demás mallas construidas para T105.

Con T525 se realizó un experimento similar con mallas. La Figura 3 muestra seis de ellas, cada una asociada a un valor de α. Como se puede notar, nuevamente se evidencia la relación $w = c_1 + 0{,}5$.

Mallas para diferentes valores de α, variando los parámetros w y c_1, en T105

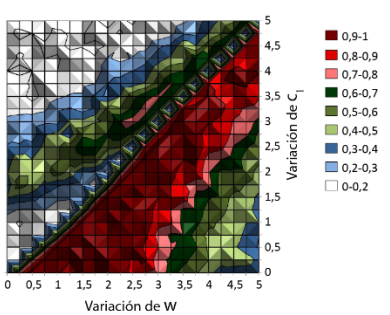

Figura 1.2: $\alpha = -0{,}50$.

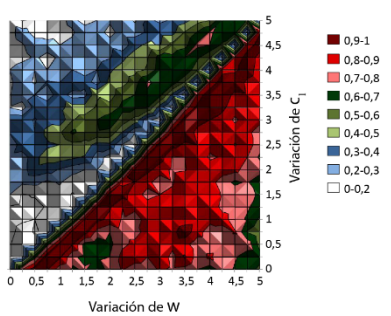

Figura 1.3: $\alpha = -0{,}25$.

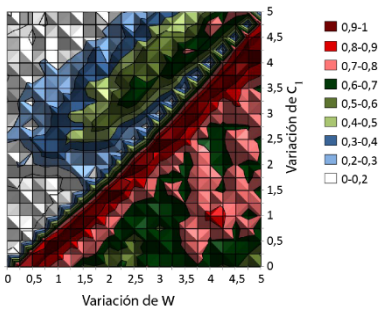

Figura 1.4: $\alpha = 0{,}25$.

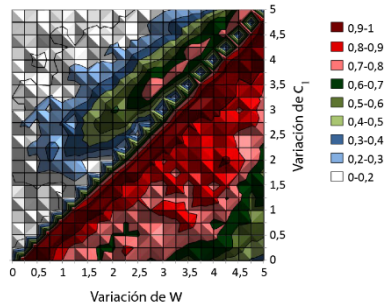

Figura 1.5: $\alpha = 0{,}50$.

En virtud de esta correspondencia se concluyó que se puede tomar cualquier pareja de parámetros (wc_1), siempre que $c_1 \in [0;4,5]$, $w \in [0,5;5]$ y $w - c_1 = 0,5$. En este caso, se seleccionóla combinación $c_1 = 1,25$ y $w = 1,75$. Por otra parte, las seis gráficas de la Figura 3 ayudan a visualizarla relación del parámetro α con el rendimiento del algoritmo. En efecto, se nota que los valores de α cercanos a cero son los que generan porcentajes de atracción más altos, en contraposición de los valores mayores que 1 o menores que -1 (este comportamiento también se tiene en las demás mallas, que fueron omitidas en el artículo por razones de espacio). Para reforzar esta conjetura, la Figura 4 muestra el rendimiento del algoritmo en ambas tablas, utilizando el juego de parámetros $c_1 = 1,25$ y $w = 1,75$, y variando el valor de α. En función de los resultados mostrados se decidió fijar $\alpha = 0,7$.

En resumen, la aplicación del algoritmo para el reporte de resultados se realizó con la combinación de parámetros $\alpha = 0,7$; $c_1 = 1,25$; $c_2 = 0,5$;$w = 1,75$; $m = 10$, $KM = 3$ y `MaxIteraSinMejora=10`.

Mallas para diferentes valores de α, variando los parámetros w y c_1, en T525.

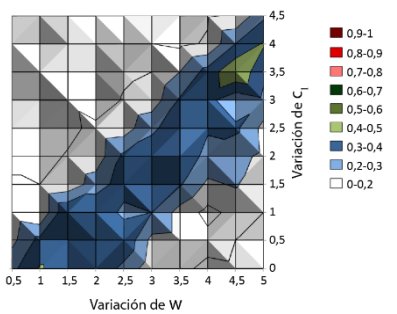

Figura 1.6: $\alpha = -2$.

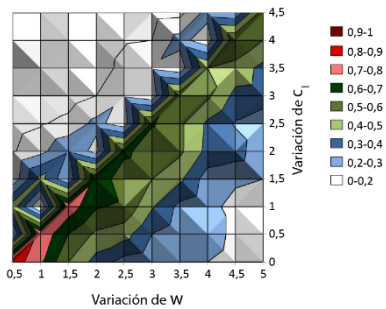

Figura 1.7: $\alpha = -1$.

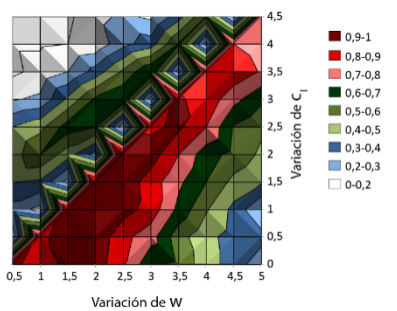

Figura 1.8: $\alpha = -0{,}5$.

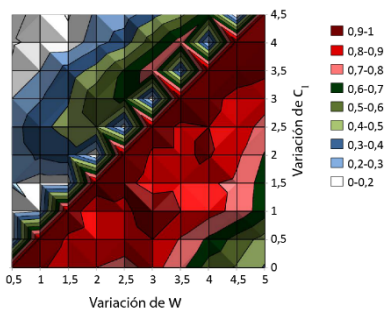

Figura 1.9: $\alpha = 0.5$.

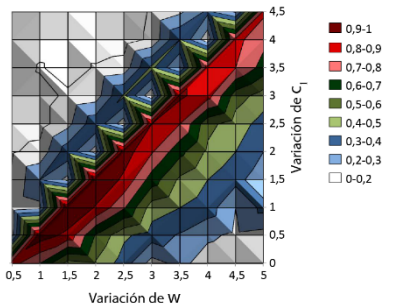

Figura 1.10: $\alpha = 1$.

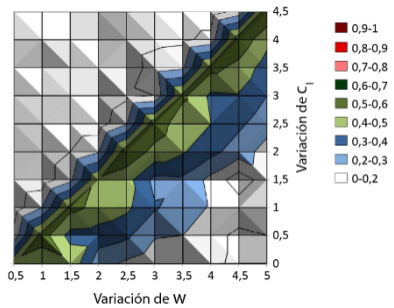

Figura 1.11: $\alpha = 2$.

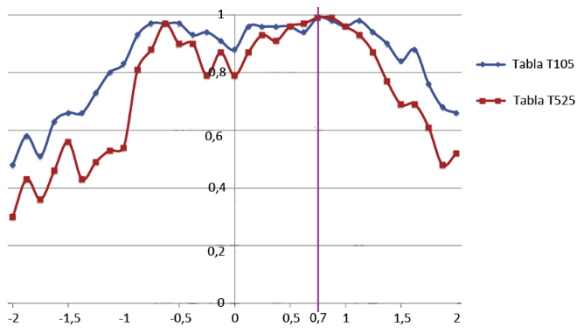

Figura 1.12: Análisis del parámetro α.

1.4 Datos utilizados y resultados

Para la prueba del algoritmo se utilizarondiez tablas extraídas de los repositorios disponibles en http://archive.ics.uci.edu/ml/ (University of California) y http://cs.joensuu.fi/sipu/datasets/ (University of Eastern Finland), las cuales se describen a continuación.

Tabla delosIris de Fisher

Tabla de 150 objetos (150 flores de tres especies: iris setosa, iris versicolor e iris virginica), que son caracterizadas en cuatro variables cuantitativas: largo y ancho del sépalo, y largo y ancho del pétalo.

Tablasde WineQuality

Consta de dos tablassobre las variantes roja y blanca del *vinho verde* (vino producido en Minho, zona al noroeste de Portugal). La primera, *winequality-red* (WQ-red), consta de 1599 muestras de vino rojo caracterizadas en 11 atributos cuantitativos. Por su parte, la tabla*winequality-white* (WQ-white) se compone de 4898 muestras de vino blanco, descritas en esos mismos atributos.

Tabla Glass

Está compuesta por 214 instancias, que corresponden a muestras de 6 clases de vidrios caracterizadas en 9 atributos cuantitativos (cantidad presente en cada muestra de Mg, K, Ca, Ba, entre otros elementos químicos).

Tablas de A-Sets

Corresponde a un conjunto de tres tablas de datos sintéticos bidimensionales (denominadas A1, A2 y A3). Cada tabla tiene un número diferente de objetos y de clases. Sin embargo, en las tres tablas cada clase contiene 150 objetos. Para el estudio se utilizaron, únicamente, las tablas A1 y A2.

Tablas de S-Sets

Corresponde a un conjunto de cuatro tablasde datos sintéticos bidimensionales (denominadas S1, S2, S3 y S4) con 5000 individuos y caracterizados con dos variables. Estas tablas están construidas con $K = 15$ clases.

La Tabla 1.2 resume las principales características de los conjuntos de datos anteriores. En particular, se indica el número n de individuos, el número de variables, el número K de clases y el valor $W(P)$ de mínima inercia intraclases que se logró determinar para cada tabla, y que se tomará como valor de referencia para calcular los porcentajes de atracción(porcentaje que representa el número de veces que el algoritmo encontró la mejor partición conocida).

Nombre de la tabla	n	**Variables**	K	$W(P)$ **de referencia**
Iris	150	4	3	0,5214
Winequality-red	1599	11	3	247,2075
Winequality-white	4898	11	3	560,4186
Glass	214	9	6	1,5704
A1	3000	2	20	4048752,5074
A2	5250	2	35	3864140,3127
S1	5000	2	15	1783523123,3735
S2	5000	2	15	2655821898,1459
S3	5000	2	15	3377914369,8714
S4	5000	2	15	3140628447,2520

Tabla 1.2: Principales características de las tablas de datos.

La Tabla 1.3 y la Tabla 1.4 muestran los resultados y tiempos obtenidos al aplicar el algoritmo de PSO a las tablas de datos, empleando diferentes números de partículas en el enjambre. Para cada combinación se ejecutó 500 veces el algoritmo para calcular el porcentaje de atracción. La notación "-" es utilizada si el algoritmo ya reportó un 100% de rendimiento para un número dado de partículas. La información dada sobre estas tablas muestra que el número de partículas influye en el porcentaje de atracción, el cual crece como función del número de partículas. Esto es consistente con la intuición de que a mayor número de agentes, mayor es la posibilidad de encontrar buenos resultados. No obstante, el número de partículas debe seleccionarse con cuidado, dado que los tiempos de ejecución también crecen dependiendo del tamaño del enjambre. Se debe buscar un equilibrio entre el rendimiento del algoritmo y los tiempos de ejecución.

Tabla	**m = 5**		**m =10**		**m = 20**		**m = 30**		**m = 50**	
	%	T(s)	%	T(s)	%	t(s)	%	T(s)	%	T(s)
Iris	100%	0,0044	-	-	-	-	-	-	-	-
WQ-red	80%	0,140	96%	0,285	99%	0,531	100%	0,799	-	-
WQ-white	100%	0,542	-	-	-	-	-	-	-	-
Glass	9%	0,014	16%	0.030	28%	0,058	40%	0,083	57%	0,141
A1	2%	1.339	6%	2,800	10%	5,492	16%	7,442	25%	11,7633
A2	5%	5,909	14%	13,41 ,414	28%	25,918	35%	34,424	38%	55
S1	78%	1,178	91%	2,305	97%	4,30	98,8%	6,161	100%	9,232
S2	94%	1,189	99%	2,315	100%	4,333	-	-	-	-
S3	37%	1,786	51%	3,718	73%	6,986	80%	10,201	93%	16,242
S4	10%	2,510	11%	5,177	10%	10,418	9,6%	14,924	12%	12,519

Tabla 1.3: Tiempos promedios en segundos y porcentajes de atracción, para varios números de partículas.

Tabla	**m = 100**		**m =150**		**m = 200**		**m = 250**		**m = 300**	
	%	T(s)	%	T(s)	%	t(s)	%	T(s)	%	T(s)
Iris	-	-	-	-	-	-	-	-	-	-
WQ-red	-	-	-	-	-	-	-	-	-	-
WQ-white	-	-	-	-	-	-	-	-	-	-
Glass	83%	0,286	94%	0,358	97%	0,465	99%	0,545	100%	0,650
A1	40%	23,13	50%	33,538	57%	44,20	64%	49,92	66%	58,27
A2	52%	109,285	58%	156,40	61%	204,347	63%	230,147	64%	291,890
S1	-	-	-	-	-	-	-	-	-	-
S2	-	-	-	-	-	-	-	-	-	-
S3	99%	29,634	100%	42,768	-	-	-	-	-	-
S4	20%	48,660	26%	36,08	35%	97,40	38%	119,456	46%	141,16

Tabla 1.4: Continuación de la información presentada en la Tabla 1.3

Finalmente, el solapamiento entre las clases es un factor que surgió en los resultados como una causa que influye en la dificultad para que el algoritmo determine el agrupamiento óptimo de los datos. En particular, la tabla S4 tiene un alto grado de intersección entre las clases y, por ende, resultó ser la tabla más complicada para determinar la clasificación óptima, obteniendo únicamente 46% de porcentaje de atracción, y empleando para ello el número más alto de partículas en el análisis (300 partículas).Este aspecto ya había sido explorado en otros análisis (Chavarría & Fallas, 2016).

1.5 Conclusiones

En el experimento, los parámetros que generaron mejores resultados cumplen la relación $w = c_1 + 0{,}5$. Por lo tanto, c_2 debe tener un valor de $0{,}5$. Por su parte, c_1 puede ser tomado en el intervalo $[0;4{,}5]$. Además, el parámetro de inercia α se debe seleccionar positivo y cercano a cero. Si bien se recomienda ajustar α para diferentes escogencias de los parámetros c_1 y w, este parámetro puede ser tomado en el conjunto $]0,1]$. Lo anterior dado que el rendimiento de la heurística decayó para $\alpha > 1$. En este estudio, el conjunto de parámetros que brindólos mejores resultados es: $w = 1{,}75$; $c_1 = 1{,}25$; $c_2 = 0{,}5$ y $\alpha = 0{,}7$.

Con respeto al algoritmo PSO se concluye que el número de partículas no es generalizable para todos los conjuntos de datos, dado que la complejidad de las tablas depende de muchas variables como: número de individuospor clasificar, número de atributos de los individuos,número de clases por construir, y la difusión y el solapamiento de las clases. Incluso, surgió evidencia de que el solapamiento entre las clases es un factor que incide significativamente para poder determinar la clasificación óptima de un conjunto de datos.

Bibliografía

[1] Alias, M., Aziz, N., Aziz, K. & Mohemmed, A (2011). Particle swarm optimization for constrained and multiobjective problems: a brief review. Memorias de International Conference on Management and Artificial Intelligence, Bali, Indonesia, 6 (1), 146-150.

[2] Babu, P. & Murty, N. (1994). Simulated annealing for selecting optimal initial seeds in the k-means algorithm. Indian Journal Pure and Applied Mathematics, 25(1-2), 85-94.

[3] Bratton, D. & Kennedy, J. (2007). Defining a standard for particle swarm optimization. Memorias del IEES SwarmIntelligenceSymposium, Hawaii, USA, 120-127.

[4] Chavarría-Molina, J; Fallas-Monge. J. (2011) Movimiento de centroides y transferencias: alternativas para construir vecinos en sobrecalentamiento simulado. Tecnología en Marcha. Edición especial. Matemática Aplicada, Mayo, Costa Rica, 30(2), 65-77.

[5] Chi-Yang, T. & I-Wei K., (2011). Particle swarm optimization with selective particle regeneration for data clustering. Elsevier. Department of Industrial Engineering and Management, Yuan Ze University, Taoyuan 320, Taiwan, ROC.

[6] Eberhart, R. & Kennedy, J. (1995). A New optimizer using particle swarm theory. Memorias de Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japón,39-43.

[7] Hassan, R., Cohanim, B. & Weck, O. (2005). A comparison of particle swarm optimization and the genetic algorithm. En memorias de First AIAA Multidisciplinary Design Optimization Specialist Conference, Austin, Texas, USA, 1-13.

[8] Kao, I.-W., Tsai, C.-Y., & Wang, Y.-C., (2007). An effective particle swarmoptimization method for data clustering. In 2007 IEEE international conferenceon industrial engineering and engineering management, Singapore (pp. 548–552).

[9] Kennedy, J. & Eberhart, R. (2001). Swarm Intelligence. AcademicPress, San Diego, CA, USA.

[10] Lima, J. & Baran, B. (2006). Optimización de enjambre de partículas aplicada al problema del Cajero viajante bi-objetivo. RevistaIberoamericana de Inteligencia Artificial, 10 (32), 67-76.

[11] Ng, M. & Wong, J. (2002). Clustering categorical data sets using tabu search techniques. Elsevier Science Ltd on behalf of Pattern Recognition Society, 35 (12), 2783-2790.

[12] Pacheco, A., Trejos, J., Piza, E. & Murillo, M. (2006). Evaluación de heurísticas de optimización combinatoria en clasificación por particiones. Revista Investigación Operacional, 27 (2), 124-128.

[13] Sarkar, M. & Yegnanarayana, B. (1996). A clusteringalgorithmusingevolutionaryprogramming. Memorias de la IEEE International Conference on Neural Network, Washington DC, USA, 1162-1167.

[14] Sedighizadeh, D. & Masehian, E. (2009). Particle Swarm optimization methods, taxonomy and applications. International Journal of Computer Theory and Engineering, 1(5), 486-502.

[15] Tan, L. (2015). A Clustering K-means Algorithm Based on Improved PSO Algorithm. Fifth International Conference on Communication Systems and Network Technologies, Gwalior, 940-944. doi:10.1109/CSNT.2015.223

[16] Trejos, J. & Murillo, A. (2004). Heuristics of Combinatorial Optimization and Applications to Data Analysis. Memorias de laI Summer School on Optimization and Numerical Analysis, Berlin, Alemania.

[17] Tsai, CW., Huang, KW., Yang, CS. et al. Soft Comput (2015). A fast particle swarm optimization for clustering. 19: 321. doi:10.1007/s00500-014-1255-3.

[18] University of Eastern Finland, School of Computing. (s.f.). Clustering datasets: Speech and Image Processing Unit [Repositorio de datos]. Recuperado desde http://cs.joensuu.fi/sipu/datasets/

[19] Xie, X., Zhang, W. & Yang, Z. (2002a). A Dissipative Particle Swarm Optimization. En Memorias de IEEE Congress on Evolutionary Computation, Honolulu,Hawai, USA, 2(1), 1456-1461.

[20] Xie, X., Zhang, W. & Yang, Z. (2002b).Adaptive Particle Swarm Optimizationon Individual Level. En Memorias de 6th International Conference onSignal Processing, Beijing, China, 2(1), 1215-1218.