ESTUDIO DE ESTRATEGIAS PARA LA ACELERACIÓN DE LA CONVERGENCIA DEL CLUSTERING MEDIANTE FUZZY C-MEANS

DIANA CAROLINA TRUJILLO ORTIZ
CARLOS SERJEIF SACRISTÁN HERNÁNDEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO - MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES
BUCARAMANGA
2016

ESTUDIO DE ESTRATEGIAS PARA LA ACELERACIÓN DE LA CONVERGENCIA DEL CLUSTERING MEDIANTE FUZZY C-MEANS

DIANA CAROLINA TRUJILLO ORTIZ
CARLOS SERJEIF SACRISTÁN HERNÁNDEZ

Trabajo de grado para optar el título de
Ingeniero Electrónico

DIRECTOR
PhD. SAID DAVID PERTUZ ARROYO
Máster en Ingeniería informática y de la seguridad

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO - MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES
BUCARAMANGA
2016

AGRADECIMIENTOS

Dedico de manera muy especial a mi padre Raul Trujillo Niño por su esfuerzo, dedicación y amor, por brindarme su apoyo incondicional, por creer fielmente en mis logros tanto personal y profesional desde niña. Gracias a Dios por regalarme un padre tan maravilloso y por ser mi bastón en cada momento.

Agradezco a mi hermano Raul Trujillo Ortiz por ser mi amigo, mi guía, mi compañía, por ser incondicional, por acompañarme en este camino y brindarme su apoyo.

Agradezco a mi madre Clara Ortiz por su lealtad, amor y por ser mi amiga. Este es nuestro sueño.

Agradezco a Yesid Murillo por su cariño, su apoyo, por regalarme las fuerzas necesarias cuando sentía que el camino se tornaba muy largo, gracias por sus palabras de aliento y sus sonrisas.

El camino no ha sido fácil pero si cuentas con personas a tu lado que creen en ti, siempre será mucho más llevadero.

Agradezco al Ph D. Said David Pertuz Arroyo, por su aporte y colaboración ya que sin estos aportes no hubiese podido culminar esta meta

Gracias a Dios por que siempre ha estado presente en cada momento de mi vida.

Diana Carolina Trujillo Ortiz

En primer lugar quiero darle gracias a Dios por ayudarme y permitirme alcanzar cada una de mis metas propuestas en mi diario vivir. A mis queridos padres por su apoyo incondicional durante el trayecto de mi carrera y mi vida, por enseñarme el camino adecuado para seguir adelante, que los obstáculos más difíciles solo se encuentran en la imaginación de personas sin rumbo. Sinceramente no sé qué habría sido de mí si no hubieran estado conmigo siempre, así que este título es más de ellos que mío.

Agradezco a mi familia quienes ayudaron a mi crecimiento profesional y me acompañaron en este trayecto de mi vida.

A mi hermanito Yustin S. por ser una de las razones de salir adelante para ser un buen ejemplo a seguir.

A ti mi amor hermosa Samira Juliana, por regalarme los mejores momentos que un hombre pudiera pedir, por tu comprensión, cariño y tus palabras de aliento para seguir adelante de la mano de Dios. Gracias por ese amor que me brindas sin condición y por estar conmigo hasta en los momentos más difíciles donde hemos luchado hombro con hombro contra el mundo.

A todos los profesores que durante mi carrera me brindaron todos sus conocimientos para mi formación como Ingeniero en esta prestigiosa y renombrada institución.

Por ultimo quiero agradecer a mis mejores amigos Harold P. Angel O. Ricardo B. y Andres M. por acompañarme en este gran trayecto de mi carrera, les deseo lo mejor de este mundo porque son las personas más estupendas que Dios me permitió conocer.

Carlos Serjeif Sacristán Hernández

CONTENIDO

Pág.

LISTA DE TABLAS

Pág.

LISTA DE ILUSTRACIONES

Pág.

NOMENCLATURA

u_{ik}	Grado de pertenencia de un dato k encontrado en un *cluster* i.
v_i	$i-ésimos$ Centroides del *cluster.*
$J_m(U,V)$	Función objetivo *FCM*.
n	Número total de datos.
X	Hace referencia a un conjunto de datos o *Dataset.*
t	Número de iteraciones que realiza un algoritmo
d	Dimensiones del *Dataset*
c	Número de *clústers.*
$D_{ik}(x_k, v_i)$	Distancia euclidiana definida como la distancia entre los $k-ésimos$ datos y los $i-ésimos$ centroides.
U	Matriz de pertenencia referida a los valores u_{ik} encontrados.
V	Conjunto de centroides referidos a los valores v_i encontrados.
m	Factor fuzzificador.
w_i	Pesos ponderados para los $i-ésimos$ Centroides del *cluster.*
eps	Criterio de parada Épsilon.
$fPDA$	Fracción de Acceso a datos parciales (*Fractional Partial Data Access*).
SU_{12}	Métrica de proporción entre el tiempo de ejecución del algoritmo 1 y el algoritmo 2.
$O(f(n))$	Orden de la complejidad.
$VDQ\%$	Métrica porcentaje de la calidad de las particiones.
$CC\%$	Métrica porcentaje de cambio de pertenencia del *cluster.*
$R_m(V)$	Criterio de optimización reformulado de $J_m(U,V)$.
$DQR_m\%$	Métrica Diferencia en la calidad de la función objetivo.

RESUMEN

TÍTULO: ESTUDIO DE ESTRATEGIAS PARA LA ACELERACIÓN DE LA CONVERGENCIA DEL CLUSTERING MEDIANTE FUZZY C-MEANS[*]

AUTORES: DIANA CAROLINA TRUJILLO ORTIZ – CARLOS SERJEIF SACRISTÁN HERNÁNDEZ[**]

PALABRAS CLAVE: ***CLUSTER, CLUSTERING, FUZZY C-MEANS (FCM), PARTICIÓN DIFUSA, CENTROIDES, CONVERGENCIA.***

CONTENIDO:

En el presente trabajo de grado se lleva a cabo el estudio de estrategias para acelerar la convergencia del *clustering* empleando *Fuzzy C-means (FCM),* con la finalidad de encontrar la mejor variante del algoritmo de *clustering FCM* que permita reducir el tiempo de cómputo, coste computacional y calidad de *clustering*. Las variantes *FCM* relacionadas con la aceleración que se seleccionaron para el estudio son *Online Fuzzy C-means* (*OFCM*), *Single Pass Fuzzy C-means* (*SPFCM*) y *Random Sampling Plus Extension Fuzzy C-means* (*RSEFCM***).** Dichas variantes son comparadas entre sí y los resultados se contrastan con el algoritmo original *FCM*. Para este análisis se utilizan tres *Dataset*s obtenidos de bases de datos públicamente disponibles, con el fin de recolectar la información necesaria en cada procedimiento de *clustering* realizado por cada algoritmo. Se observó que al comparar las métricas de cómputo con cada uno de los *Datasets* con respecto al Speed up (aceleración), la variante de (*RSEFCM)* tuvo el mejor desempeño. Para el análisis de la calidad de las particiones se observó que variante de (*RSEFCM)* presenta una alta eficacia y confiabilidad en los resultados. Por último, respecto a la complejidad de los algoritmos se pudo comprobar que el coste computacional tiene un comportamiento lineal.

[*] Proyecto de Grado.

[**] Facultad de Ingenierías Fisico-Mecanicas. Escuela de Ingeniería eléctrica, Electrónica y Telecomunicaciones. Director: Said David Pertuz Arroyo.

SUMMARY

TITLE: STUDY OF STRATEGIES FOR THE ACCELERATION OF THE CONVERGENCE OF CLUSTERING USING FUZZY C-MEANS[*]

AUTORS: DIANA CAROLINA TRUJILLO ORTIZ – CARLOS SERJEIF SACRISTÁN HERNÁNDEZ[**]

KEYWORDS: CLUSTER, CLUSTERING, FUZZY C-MEANS (FCM), FUZZY PARTITION, CENTROIDS, CONVERGENCE.

CONTENT:

In the present work, we perform a comparative study of strategies to accelerate the convergence of *clustering* using *Fuzzy C-means (FCM),* in order to find the best implementation of the *FCM clustering* algorithm in terms of computational cost. The studied *FCM* variants related to the acceleration of *clustering* were *Online Fuzzy C-means (OFCM),* and *Single Pass Fuzzy C-means (SPFCM),* and *Ramdom Sampling plus Extension Fuzzy C-means (RSEFCM)* which are compared against the original *FCM* algorithm using three different databases in order to collect the necessary information about each *clustering* algorithm. I our experiments for assessing the speed up with respect to the classic FCM, we found that RSEFCM had the highest performance. With respect to the quality of the obtained partitions, RSEFCM yielded the more accurate and robust results among the studied versions of FCM. As for the complexity of the algorithms, experimental tests showed a linear behavior in the computational cost.

[*] Degree Work.

[**] Faculty of physicomechanical Engineering. Shool of Electric, Electronic and Telecomunications. Project Director: Said David Pertuz Arroyo.

INTRODUCCIÓN

Una de las más antiguas funciones cerebrales que el ser humano ha desarrollado con el pasar del tiempo es el proceso de agrupamiento. Filósofos y científicos reflexionaban sobre el funcionamiento del agrupamiento, a través de cuestionamientos acerca de cómo era posible percibir la realidad, enfrentándose al problema de la descripción de la materia y de las formas mediante atributos o características. Según hechos históricos pasados y planteamientos filosóficos como los hechos por Platón en su obra "El mito de la caverna", es posible afirmar que el ser humano tiene la capacidad de identificar y analizar las características de los objetos, logrando un agrupamiento basado en dichas mediciones para organizar la información[1].

Tiempo atrás, la información se recopilaba y clasificaba manualmente debido a la falta de avances técnicos que permitieran agilizar el proceso. En la actualidad dicha información ha ido aumentando, lo cual ha incrementado el costo de la clasificación a mano. De esta manera se ha descubierto la necesidad de crear nuevos mecanismos o métodos que permitan una organización a bajo costo. Con el transcurrir del tiempo, se ha logrado obtener avances tecnológicos y la capacidad de cómputo ha mejorado, lo que ha permitido crear algoritmos de agrupación conocidos como algoritmos de *clustering.* Específicamente, el *clustering* es una técnica que realiza divisiones de datos en grupos de objetos similares, con el fin de recopilar, clasificar y organizar la información de una forma más útil, clara y sencilla.

Actualmente el manejo y la gestión de bases de datos adquieren mayor relevancia, a medida que se realiza un análisis profundo y detallado de ellas. Por

[1] S. BECA COFRE, "Clustering difuso con selección de atributos," p. 100, 2007.

este motivo es necesario contar con técnicas de *clustering* que permitan obtener información que antes no se tenía en cuenta. En este campo, se han desarrollado diferentes métodos y algoritmos, tales como *K-means, Fuzzy C-means, Gaussian Mixtures,* entre otros[2].

Cada algoritmo de *clustering* está diseñado para dar solución a diferentes casos de agrupamientos y se pueden clasificar como algoritmos jerárquicos, probabilísticos y particionales. El estudio de estos algoritmos tiene distintas aplicaciones. Un ejemplo es el desarrollo de algoritmos de *clustering* para la clasificación y agrupación de información de motores de búsqueda como *Google* o *Yahoo!,* Los cuales facilitan la búsqueda de información en internet[3].

[2] J. WU, Advances in K-means Clustering, vol. 53, no. 9. china, 2013.
[3] J. W. MARÍN, a ,BRANCH B, "Aplicación de dos nuevos algoritmos para agrupar resultados de búsquedas en sistemas de catálogos públicos en línea (OPAC).," *Rev. Interam. Bibl.*, vol. 31, pp. 47–65, 2008.

1. PLANTEAMIENTO DEL PROBLEMA

Uno de los problemas más importantes que tiene que ver con los algoritmos de *clustering* es la eficiencia computacional, siendo de carácter prioritario reducir el tiempo de cómputo mientras se mantienen o mejoran los niveles de desempeño de un algoritmo. El presente proyecto tiene como propósito estudiar estrategias de aceleración de algoritmos de *clustering.* En particular, el estudio se centrará en el algoritmo *Fuzzy C-means (FCM),* el cual es una técnica difusa bastante conocida y que en los últimos años ha cobrado importancia*,* siendo una versión difusa del conocido algoritmo clásico *C-Means* [4].

1.1. DESCRIPCIÓN DE LA INVESTIGACIÓN

Este informe presenta el estudio comparativo de diferentes estrategias para acelerar el algoritmo *Fuzzy C-means.* Inicialmente se realiza una investigación sobre las diferentes modificaciones que se han realizado sobre el algoritmo original, previamente se procede a escoger tres variantes de este algoritmo.

1.2. OBJETIVO GENERAL

Estudiar y comparar estrategias para la aceleración de la convergencia del *clustering* mediante el algoritmo *Fuzzy C-means.*

[4] S. Ghosh and S. K. S. Dubey, “Comparative analysis of k-means and fuzzy c-means algorithms,” *Ijacsa*, vol. 4, no. 4, pp. 35–38, 2013.

1.3. OBJETIVOS ESPECIFICOS

El cumplimiento del objetivo general del trabajo de grado comprende:

1. Estudiar y seleccionar estrategias para la aceleración de algoritmos *clustering*.
2. Implementar las estrategias seleccionadas para el *clustering* mediante *Fuzzy C-means* para un conjunto de datos en C++.
3. Comparar el desempeño de las estrategias implementadas tomando como métrica el tiempo de cómputo y la complejidad computacional de las implementaciones de los algoritmos.

2. FUNDAMENTO TEÓRICO

En este capítulo se describen, de forma breve, los conceptos principales que abarcan la teoría necesaria para la comprensión del presente trabajo de grado en la modalidad de investigación. Específicamente, Se presentan algunos fundamentos sobre el *clustering.* Primero se presentan los diferentes tipos de algoritmos de *clustering,* luego se presenta el algoritmo de *Fuzzy C-means*; por último, se mencionarán los algoritmos de prueba escogidos para ser comparados por la diferencia de agrupaciones entre cada uno[5].

2.1. MÉTODOS DE *CLUSTERING*

Existen una gran variedad de métodos de *clustering* los cuales, según su principio de funcionamiento, se pueden clasificar en los siguientes tres grupos:

2.1.1. Métodos jerárquicos. Este método consiste en agrupar o separar *clusters* con el objetivo de formar uno o varios *clusters* nuevos, permitiendo la construcción de un árbol de clasificaciones según los *clusters* creados lo que se conoce como dendogramas. Los métodos jerárquicos se dividen en dos métodos los cuales son[6]

Métodos acumulativos ascendentes, que van continuamente uniendo grupos en cada paso; y métodos divisores descendentes, que van separando en grupos cada vez más diminutos, algunos ejemplos son:

[5] S. GHOSH and S. K. S. DUBEY, “Comparative analysis of k-means and fuzzy c-means algorithms,” Ijacsa, vol. 4, no. 4, pp. 35–38, 2013.
[6] “Métodos Jerarquicos de Analisis Cluster,” 2014.

2.1.1.1. Método de la distancia mínima: La principal función de este método es considerar la distancia entre los *clusters* como la distancia más pequeña entre los datos más próximos. Por este motivo el método es considerado como espacio-contractivo, el cual tiende acercar los datos más de lo que muestran sus distancias propuestas inicialmente. Esta técnica no ha tenido buenas críticas por considerarse muy sensible en el caso cuando hay datos perturbadores entre *clusters* muy bien diferenciados, presenciado en casos de ruido[7].

2.1.1.2. Método de la distancia máxima: Este procedimiento es espacio-dilatante, es decir, tiende a dividir los datos en una medida más alta que la mostrada por la desemejanza inicial. Es muy poco usado puesto que presenta inconvenientes a la hora de ejecutar y alarga considerablemente el proceso, lo que genera agrupaciones enlazadas[8].

2.1.1.3. Método de la media: Aunque los métodos anteriores poseen excelentes propiedades teóricas, tienen la particularidad de distorsionar las medidas iniciales de disimilaridad, ampliando respectivamente la métrica. Una solución óptima para este problema fue considerado por Sokal y Micheber[9], quienes propusieron el método de Group Average[10]. Esta técnica propone utilizar como distancia entre un grupo I y un individuo j la media de las distancias entre los individuos del grupo I y el individuo j.

2.1.1.4. Método de la mediana: Es un método basado en centroides, también conocido como método del centroide no ponderado el cual es espacio-conservativo y no tiene en cuenta los tamaños de los *clusters*. Por lo tanto no resulta ser invariante ante transformaciones monótonas de dicha distancia empleada. Siendo este uno de los más usados en la parte práctica; es muy similar

[7] C. P. LOPEZ, *Mineria de datos: técnicas y herramientas*, Segunda ed. España, 2008.
[8] Ibid
[9] P. LEMEY, M. SALEMI, and A. VAMDAMME, The phylogenetic handbook, SECOND EDI. USA: 2009, 2009.
10 P. RAGHAVAN Christopher D. Manning, “Group-average agglomerative clustering,” 2009. [Online]. Available: http://nlp.stanford.edu/IR-book/html/htmledition/group-average-agglomerative-clustering-1.html.

al método de la media aunque siendo más selectivo en la determinación de los niveles de agrupación[11].

2.1.2. Métodos Particionales: Este método tiene como objetivo obtener una partición de los datos en *clusters,* es decir, crear *k* agrupaciones de dichos datos. Este método se diferencia a los jerárquicos en que los *clusters* asignados son disjuntos, donde los *k clusters* no tienen ningún elemento en común o expresado de otra forma, la intersección entre los *k* grupos disjuntos es vacío. Los métodos particionales están clasificados en particiones duras y particiones difusas. Una partición difusa se caracteriza por asignar a cada dato un valor de pertenencia entre cero y uno utilizando funciones de pertenencia, lo que permite que un dato pueda pertenecer a diferentes *clusters* en cada iteración, mientras que una partición dura se caracteriza por que cada dato pertenece únicamente a un *cluster* en específico. Esta investigación está basada en el estudio del algoritmo Fuzzy C-means el cual trabaja con particiones difusas[12].

2.1.3. Método Probabilísticos: Consiste en encontrar el grupo de *clusters* más probables, donde los datos tienen cierta probabilidad de pertenecer a un *cluster.* La base de un *clustering* probabilístico está basado en el modelo estadístico llamado *finite mixtures* o mezcla de distribuciones. Este método trata de encontrar los individuos que pertenecen a una misma distribución, buscando zonas de mayor concentración. También es conocido como métodos de búsqueda de la densidad.

Entre los algoritmos más ampliamente utilizados, se encuentran el algoritmo de *Fuzzy C-means* (*FCM*) pertenecientes a los métodos particionales de partición

[11] M. Gallardo, “Aplicación De téccnicas De Clustering Para La Mejora Del Aprendizaje,” 2009.
[12] P. Larra, “Clustering,” pp. 1–11.

dura y difusa respectivamente. Los detalles del algoritmo FCM se presentan en la siguiente sección[13].

2.2. ALGORITMO FUZZY *C*-MEANS (*FCM*)

Muchas veces encontramos en la vida cotidiana que un elemento está lo suficientemente cerca de dos *clusters,* de tal manera que es muy difícil saber en cual *cluster* colocarlo. El algoritmo de *Fuzzy C-means,* está diseñado con el objetivo de solucionar tales inconvenientes presentados en un *clustering.*

El algoritmo *Fuzzy C-means* asigna a cada elemento un valor de pertenencia y de esta forma un elemento específico puede pertenecer parcialmente a más de un *cluster.* Esta técnica se basa en el principio del algoritmo clásico *C-means*, con la única diferencia de que la partición del conjunto de elementos realizada por el algoritmo no es dura sino suave[14].

Formalmente, el grado de pertenencia al objeto o dato k respecto al *cluster* i está definido como u_{ik}, teniendo en cuenta las siguientes restricciones:

$$u_{ik} \in [0,1], \ 1 \leq i \leq c, \ 1 \leq k \leq n \quad (1)$$

$$\sum_{i=1}^{c} u_{ik} = 1, \qquad 1 \leq k \leq n \quad (2)$$

$$\sum_{k=1}^{n} u_{ik} > 0, \qquad 1 \leq i \leq c \quad (3)$$

[13] M. Gallardo, Op cit.
[14] J. C. M. ROJAS DIAZ, Jerónimo; Chavarro Porras and R. Laverde, "Técnicas De Lógica Difusa Aplicadas a La Minería De Datos," *Sci. Tech.*, vol. XIV, no. x, pp. 1–6, 2008.

Donde n es el número de datos totales, c el número de *clusters* y u_{ik} es el grado de pertenencia de un dato k encontrado en un *cluster* i.

El algoritmo *FCM* tiene como objetivo minimizar una función objetivo, y generar un cálculo óptimo de los centroides asignando a cada dato un valor de pertenencia. Suponiendo que los datos de entrada corresponden a un vector de características x_k, la función objetivo J_m, los valores de pertenencia u_{ik}, y los centroides v_j, son definidos de la siguiente manera:

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m}(x_k, v_i) \tag{4}$$

$$u_{ik} = \frac{D_{ik}(x_k, v_i)^{\frac{1}{1-m}}}{\sum_{j=1}^{c} D_{jk}\left(x_k, v_j\right)^{\frac{1}{1-m}}} \tag{5}$$

$$v_i = \frac{\sum_{j=1}^{n}(u_{ij})^m x_j}{\sum_{j=1}^{n}(u_{ij})^m} \tag{6}$$

Donde $m > 1$ es el factor fuzzificador, U es la matriz de pertenencia referida a los valores de u_{ik}, V es el conjunto de centroides de acuerdo a los valores v_i centroides encontrados y $D_{ik}(x_k, v_i)$ es el cuadrado de la distancia entre los $k - ésimos$ datos y los $i - esimos$ centroides. En este caso se utilizó como métrica la distancia euclidiana[15]. El algoritmo FCM trabaja de forma iterativa repitiendo los cálculos de las ecuaciones (4)-(6), tal como se muestra en la Ilustración 1.

[15] M. GALLARDO, "Aplicación De técnicas De Clustering Para La Mejora Del Aprendizaje," 2009.

Ilustración 1. Pseudocódigo Algoritmo *FCM*.

Pseudocódigo**: **Algorithm FCM

Input: $X, c, m \quad Where\ X\ is\ the\ Dataset$
Output: U, V
$Initialize\ V \quad (set\ of\ cluster\ centers)$
$\boldsymbol{while}\ max\ \ 1 \leq k \leq c \left\{ \left|\left| v_{k,new} - v_{k,old} \right|\right|^2 \right\} > \in$ **do**

$$u_{ik} = \frac{D_{ik}(x_k, v_i)^{\frac{1}{1-m}}}{\sum_{j=1}^{c} D_{jk}(x_k, v_j)^{\frac{1}{1-m}}} \quad , \forall i, k$$

$$v_i = \frac{\sum_{j=1}^{n} (u_{ij})^m x_j}{\sum_{j=1}^{n} (u_{ij})^m}, \quad \forall i$$

Fuente: T. C. HAVENS, et al, "Fuzzy c-Means algorithms for very large data," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, 2012.

2.3. VARIANTES DE LA ACELERACIÓN DEL *FCM*.

En los últimos años se ha presentado un incremento en el volumen de datos en las distintas ramas de la industria, por ello se hace más compleja la búsqueda de dichos datos, Una solución a este problema se hace por medio de la aceleración del *clustering*.

En este trabajo de grado se estudia el tiempo de ejecución del algoritmo de *Fuzzy C-means* (*FCM*), comparándolo con 3 algoritmos variantes del *FCM* que se describen a continuación.

2.3.1. Single Pass Fuzzy C-means (SPFCM). El algoritmo *SPFCM* es una modificación o variante del algoritmo *FCM* y tiene como finalidad realizar el *clustering* de dicho conjunto de datos escaneándolos una sola vez[16]. Esta

[16] T. C. HAVENS, J. C. BEZDEK, C. LECKIE, L. O. HALL, and M. PALANISWAMI, "Fuzzy c-Means algorithms for very large data," IEEE Trans. Fuzzy Syst., vol. 20, no. 6, pp. 1130–1146, 2012.

estrategia fue desarrollada con el fin de analizar conjuntos de datos extensos y obtener un *clustering* igual que *FCM* sin la necesidad de analizar todos los datos, es decir reducir de cierta manera el tiempo de ejecución del algoritmo.

El algoritmo realiza una división del conjunto de datos (*Dataset*) en *PDA* -por sus siglas en inglés "*Partial Data Accesses*"-, los cuales tienen un porcentaje de los datos. El número de *PDA* dependerá de cuantos datos deseemos cargar cada vez, y el tamaño de cada *PDA* será igual a n veces el parámetro *PDA* fraccionado (*fPDA*), donde n es igual al número total de datos.

Después de obtener el primer *PDA*, los datos se agrupan en c particiones utilizando *FCM*. Luego del *clustering,* los datos en la memoria son cargados en c puntos ponderados debido a su asociación de pesos de cada dato, el cual es calculado sumando las muestras de membrecía en un *cluster.* En cada *PDA* se cargan en memoria nuevos puntos únicos y son agrupados junto con los anteriores puntos ponderados c obtenidos en el *clustering* anterior. Lo anterior recibe el nombre de PDC por sus siglas en inglés "*Partial Data Clustering*". El pseudocódigo de este algoritmo se encuentra en la Ilustración (3).

Este algoritmo realiza un proceso diferente al *FCM* el cual consiste en asignar a cada elemento o dato x_i un peso w_i modificando la función objetivo y el cálculo de los centroides de la siguiente manera[17]:

$$J_{mw}(U,V) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m} * w_k * D_{ik}(x_k, v_i) \tag{7}$$

$$v_i = \frac{\sum_{j=1}^{n} w_j * u_{ik}^{m} * x_j}{\sum_{j=1}^{n} w_j * u_{ik}^{m}} \tag{8}$$

[17] T. C. HAVENS, et al, "Fuzzy c-Means algorithms for very large data," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, 2012.

Esta nueva versión ponderada se le conoce como *Weighted Fuzzy C-means (WFCM).* Esta variante calcula puntos ponderados o pesos ponderados para cada centroide después de ser obtenidos para la primera *PDA* y son definidos de la siguiente forma (ver Ilustración 2):

$$w_i' = \sum_{j=1}^{n} (u_{ij}) w_j \,, \quad 1 \le i \le c \tag{9}$$

Es necesario tener en cuenta que inicialmente a todos los datos se les asigna un peso de 1.La representación de estos centroides ponderados indica la información de la partición del conjunto de datos desde la primera *PDA*. Finalmente el valor del conjunto de centroides V obtenido en la primera *PDA* es usado como condición inicial para V en la segunda *PDA* y así sucesivamente se repite hasta que se puedan procesar todos los datos y se obtengan un conjunto de centroides final[18].

Ilustración 2. Pseudocódigo Algoritmo W*FCM*.

Pseudocódigo:

Algorithm WFCM to minimize $J_{mw}(U,V)$

Input: $X, c, m, \boldsymbol{w}$, (initial V)
Output: U, V
$If\ V\ is\ not\ initialized, initialize\ V$,
$\boldsymbol{while}\ max\ \ 1 \le i \le c \left\{ \left|\left| v_{i,new} - v_{i,old} \right|\right|^2 \right\} > \in$ **do**

Calculate U with Eq. (5)

$$v_i = \frac{\sum_{j=1}^{n} w_j (u_{ik}^m)\, x_j}{\sum_{j=1}^{n} w_j (u_{ik}^m)}, \quad \forall i$$

Fuente: T. C. HAVENS, et al, “Fuzzy c-Means algorithms for very large data,” *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, 2012.

[18] Ibid

Ilustración 3. Pseudocódigo Algoritmo *SPFCM*.

Pseudocódigo:

Algorithm SPFCM to approximately minimize $J_{mw}\ (U, V)$

Input: X, c, m, n_s

Output: V

Load $X\ as\ n_s\ sized\ randomly\ chosen\ subsets$

$X = \{X_1, X_2, \dots, X_S\}$

1. $w = 1_{n_s}$

2. $U, V = WFCM(X_l, c, m, \boldsymbol{w})$

for $l = 2\ to\ s\ \boldsymbol{do}$

3.

$$w_i' = \sum_{j=1}^{n} (u_{ij}) w_j\ , \quad 1 \le i \le c$$

4. $w = \{w' \cup 1_{n_s}\}$

5. $U, V = WFCM(V \cup X_l\}, \mathrm{c}, \mathrm{m}, \mathrm{w}, \mathrm{V})$

Fuente: T. C. HAVENS, et al, “Fuzzy c-Means algorithms for very large data,” *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, 2012.

2.3.2. Online Fuzzy C-means (OFCM). El algoritmo *OFCM* es una variante similar al *SPFCM* pero realiza un procedimiento que logra diferenciarlo de este. Al momento de analizar Datasets con determinadas dimensiones, es posible observar que algunos no están diseñados para una randomización, por ello el *Dataset* es dividido en *PDA*s de igual forma en que se realiza mediante *SPFCM*. Los centroides de cada *PDA* son inicialmente escogidos mediante *FCM* y sus pesos son calculados usando la ecuación (9). A pesar de que sus procedimientos

son iguales al inicio, el *OFCM* no agrega los centroides ponderados a una siguiente *PDA*, sino que los guarda con el fin de crear un conjunto de centroides ponderados al procesar todas las *PDA*s. Este conjunto es analizado por el W*FCM* como un solo *Dataset* y nuevamente se obtiene un conjunto de centroides finales[19]. En la Ilustración (4) se especifica el pseudocódigo del algoritmo.

En este caso los centroides de la *PDA* anterior se usan como inicialización de otros *PDA*s. El conjunto de centroides final para W*FCM* es creado por la siguiente ecuación:

$$V_i = \frac{\sum_{j=1}^{n_c} w_j (u_{ij})^m \; x'_j}{\sum_{j=1}^{n_c} w_j (u_{ij})^m} \;\; 1 \leq i \leq c, \qquad x'_j \in X' \tag{10}$$

x'_j Puede ser una muestra original o un centroide ponderado y X' es la unión de las muestras originales y todas las muestras ponderadas o centroides[20].

Al igual que el algoritmo *SPFCM* w_j es calculada a partir de la ecuación (9) y se procede a realizar un número de iteraciones aleatorias mientras finaliza de procesar cada *PDA*[21].

[19] P. HORE, L. O. et al. “Online fuzzy C means,” *Annu. Conf. North Am. Fuzzy Inf. Process. Soc. - NAFIPS*, pp. 1–5, 2008.
[20] Ibid
[21] Ibid

Ilustración 4. Pseudocódigo Algoritmo *OFCM*.

Pseudocódigo:

Algorithm: OFCM to approximately minimize $J_{mw}(U,V)$

Input: X, c, m, n_s
Output: V
Load $X\ as\ n_s\ sized\ subsets, X = \{X_{1,}X_2, \dots, X_S\}$,
Where $X_i = \{X_{(i-1)}n_s + 1, \dots, X_i n_s\}$

1. $U_{l,}V_1 = wFCM(X_{1,}c, m, 1n_s)$

for $l = 2\ to\ s\ do$

2. $\boldsymbol{U_l, V_l} = wFCM(X_l, c, m, 1n_s V_{l-1})$
3. $wl = \sum_{j=1}^{n_s}(U_l)_j, l = 1, \dots, s$
4. $wFCM(\{V_{1,} \cup \dots \cup V_s\}, \mathrm{c}, \mathrm{m}, \{w_1 \cup \dots \cup w_s\})$

Fuente: T. C. HAVENS, et al, "Fuzzy c-Means algorithms for very large data," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, 2012.

2.3.3. Random Sampling Plus Extension Fuzzy C-means (RSEFCM). El algoritmo *RSEFCM* tiene como objetivo minimizar el conjunto de datos tomando una muestra aleatoria y sobre esta emplear el algoritmo *FCM*.

Este algoritmo facilita la aceleración reduciendo el tamaño de la muestra, dichas muestras son iguales a las $fPDA * n$, donde $fPDA$ es menor o igual a 0.5, y es un parámetro definido por los algoritmos *SPFCM* y *OFCM*, donde n es igual al número total de datos[22].

[22] J. K. PARKER, L. O. HALL, AND J. C. BEZDEK, "Comparison of Scalable Fuzzy Clustering Methods," vol. 1, no. 5, pp. 10–15, 2012.

Ilustración 5. Pseudocódigo Algoritmo *RSEFCM*.

Pseudocódigo:
Algorithm: RSEFCM minimiza aproximadamente $J_m\,(U,V)$
Input: X, c, m **Output:** $\boldsymbol{U}, V$ **1.** $Sample\ n_s\ objects\ from\ X\ without\ replacement,$ $denoted\ X_s$ **2.** $\boldsymbol{U_s}, \boldsymbol{V} = LFCM(X_s, c, m)$ **3**. $Extend\ the\ partition\ (U_s, V)\ to\ X, \forall \mathbf{x_i} \notin X_S,$ $using\ Eq.(5),\ giving\ (U,V).$

Fuente: T. C. HAVENS, et al, “Fuzzy c-Means algorithms for very large data,” *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, 2012.

3. ANÁLISIS DE DESEMPEÑO DE LOS ALGORITMOS

En cada implementación de algoritmos de *clustering,* es de vital importancia evaluar los *clusters,* para determinar la calidad y la eficiencia computacional que provee aplicar dicha técnica a un conjunto de datos determinada.

Para ello existen ciertos requisitos que deben tenerse en cuenta para la validación de *clusters*:

I. Analizar y estudiar cómo los datos tienden a realizar agrupaciones, para así determinar la existencia de una estructura no aleatoria entre los mismos.
II. Establecer el número correcto de *clusters.*
III. Evaluar el desempeño de los resultados de un *clustering* sin tener referencia de información externa.

Las medidas de desempeño son aplicadas con el fin de juzgar ciertos aspectos sobre la validez de cada *cluster* realizado. Generalmente son clasificados en:

- Supervisados (Validación externa): Miden el grado de conciencia de la distribución del *cluster* encontrada por el algoritmo de *clustering* con alguna estructura externa.
- No-supervisados (Validación interna): Miden la bondad de la organización del *clustering* sin referencia a información externa[23].

Para evaluar la velocidad y calidad de los algoritmos se utilizara el siguiente indicador:

[23] A. VILLAGRA and G. LEGUIZAMÓN, “Metaheurísticas aplicadas a Clustering,” universidad nacional de san luis, 2009.

3.1. Relative Speedup ($SU_{1,2}$)

Este indicador calcula la relación entre los tiempos de ejecución de dos algoritmos. Si T_1es el tiempo utilizado por el algoritmo que tomamos de referencia (algoritmo 1) y T_2 es el tiempo para el algoritmo 2, la aceleración (SU_{12}) del algoritmo 2 con respecto al algoritmo 1 es[24]:[25]

$$SU_{1,2} = \frac{T_1}{T_2} \tag{11}$$

Si el tiempo de ejecución del algoritmo 1 es de $750\ [ms]$ y el algoritmo 2 demora $150\ [ms]$, la relación de proporción es igual a 5, es decir, el algoritmo 2 es cinco veces más rápido con respecto al algoritmo 1. Para este trabajo de grado el algoritmo que se toma de referencia es el *FCM* clásico (T_1).

Para hacer análisis más profundos de algoritmos y poder entender su funcionamiento óptimo en cuanto a un problema, es necesario utilizar herramientas de comparación como la complejidad computacional. El análisis de estos algoritmos sirve para comparar dos o más algoritmos distintos, predecir su comportamiento en cuanto a circunstancias externas y ajustar sus parámetros para obtener un mejor resultado. En un análisis se pueden calcular distintas métricas de comparación de algoritmos, pero en este trabajo de grado se escogió el tiempo de ejecución y la complejidad computacional ya que son adecuadas para realizar un análisis en cuanto a la rapidez de los mismos[26].

[24] J. K. PARKER AND L. O. HALL, "Accelerating fuzzy-c means using an estimated subsample size," *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 5, pp. 1229–1244, 2014
[25] J. K. PARKER, L. O. HALL, AND J. C. BEZDEK, "Comparison of Scalable Fuzzy Clustering Methods," vol. 1, no. 5, pp. 10–15, 2012.
[26] J. K. PARKER AND L. O. HALL, Op cit.

3.2. TIEMPO Y COMPLEJIDAD COMPUTACIONAL

El análisis de la complejidad computacional se basa en distinguir si un algoritmo es mucho más complejo que otro, de manera teórica, con la cantidad de operaciones del tamaño de la entrada (n). Esto se hace mediante su ejecución y evaluando que tan eficaz es para resolver un problema. Por lo tanto, un algoritmo se considera más eficiente cuanto menos complejo sea.

La complejidad computacional se puede determinar de las siguientes formas:

- Empírica o experimental: Está basada en la medición directa de resultados reales.
- Teórica: Es de forma flexible y muy económica, depende de la máquina y de los resultados exactos.

Para conocer la complejidad computacional es importante saber el tiempo de ejecución de un algoritmo en función de N, donde N en este caso es el número de datos de un *Dataset* (n). Así, por ejemplo, el tiempo de ejecución $T(N)$ de la línea de comando de la ecuación (12) puede obtenerse como se observa en la ecuación (13)[27].

$$S_1; for\ (int\ i = 0; < N;\ i++)\{S_2\} \tag{12}$$

$$T(N) = T_1 + T_2 * N \tag{13}$$

[27] J. A. MAÑAS, "Análisis de Algoritmos: Complejidad," 1997. [Online]. Available: http://www.lab.dit.upm.es/~lprg/material/apuntes/o/.

Donde:

T_1: Tiempo de ejecución de la sucesión S_1

T_2: Tiempo de ejecución de la sucesión S_2

En general los algoritmos incluyen alguna sentencia condicional, la cual al ejecutarse depende de los datos que se presentan. Por lo tanto hace que sea un rango de valores. Si N tiende a infinito, su comportamiento es asintótico.

Para evaluar si un algoritmo posee una alta complejidad de cálculo es necesario analizar la potencia de los algoritmos independientemente de la máquina que los ejecuta o la habilidad del programador.

Un conjunto de funciones que comparten un mismo comportamiento asintótico se les conoce como un "Orden de complejidad". El cual se denomina $O()$, y en un algoritmo se puede presentar una gran cantidad de estos[28].

Para definir que una función $T(n)$ es de orden $O(f(n))$, deben existir unas constantes n_0 y c que cumplan la siguiente condición:

$$T(n) \leq c * f(n) \quad \forall \; n \geq n_0 \tag{14}$$

$O\big(f(n)\big)$ Está definida como el orden de complejidad de una función $f(n)$. En la Tabla 1 se muestra los diferentes tipos de orden de complejidad más sencillos que se pueden encontrar en un algoritmo[29]:

[28] Ibid
[29] "Tiempo de ejecucion. Notaciones para la Eficiencia de los Algoritmos," 19AD. .

Tabla 1. Orden de complejidad

$O(1)$	Orden Constante
$O(log\ n)$	Orden Logarítmico
$O(n)$	Orden Lineal
$O(n\ log\ n)$	Orden Algorítmico
$O(n^2)$	Orden Cuadrático
$O(n^a)$	Orden Polinomial $(a > 2)$
$O(a^n)$	Orden Exponencial $(a > 2)$
$O(n!)$	Orden Factorial

Fuente: A. Villagra and G. Leguizamón, "Metaheurísticas aplicadas a Clustering," universidad nacional de san luis, 2009.

Lo ideal a la hora de calcular la complejidad computacional es encontrar el que cumpla con el menor orden. Si un programa se ejecuta pocas veces la complejidad computacional puede ser irrelevante puesto que esta se analiza para programas con mayor complejidad en su programación[30].

En la siguiente tabla se aprecia la complejidad computacional de los algoritmos estudiados:

Tabla 2. Complejidad en tiempo y espacio

Algoritmo	*Tiempo*	*Espacio*
W*FCM*,*FCM*	$O(tc^2dn)$	$O((d+c)n)$
SPFCM	$O(tc^2dn)$	$O((d+c)(n/s))$
OFCM	$O(tc^2dn)$	$O\big((d+c)(n/s)\big)+cs)$
RSEFCM	$O(tc^2dn/s)$	$O\big((d+c)(n/s)\big)$

Fuente: J. C. M. Rojas Díaz, Jerónimo; Chavarro Porras and R. Laverde, "Técnicas De Lógica Difusa Aplicadas a La Minería De Datos," *Sci. Tech.*, vol. XIV, no. x, pp. 1–6, 2008.

[30] J. A. MAÑAS, "Análisis de Algoritmos: Complejidad."

La Tabla 2 representa el análisis de la complejidad en tiempo y espacio de los algoritmos de *clustering*. La complejidad en tiempo corresponde al tiempo de cálculo necesario para ejecutar las operaciones de un algoritmo y la complejidad en espacio representa la cantidad de memoria que un algoritmo contiene y utiliza durante su ejecución. Donde n es el número datos de d dimensiones, $X \in R^d$; c es el número de *clusters*; t es el número de iteraciones necesarias para la terminación; y s es el número de subconjuntos de X que se divide por muestreo aleatorio[31].

3.3. CALIDAD DE *CLUSTERS*

3.3.1. Calidad de particiones (VDQ%). Esta métrica está diseñada para anexar una medición a la variación de los centroides predichos de un algoritmo comparado con el algoritmo base, el cual puede ser usado para comparar uno o dos algoritmos, este se calcula de la siguiente forma[32] [33]

$$VDQ\% = \left(\frac{\sum_{j=1}^{c} \left\| V'_{ij} - V_{avgj} \right\|}{q * \sum_{j=1}^{c} \left\| V_{avgj} \right\|} \right) * 100 \qquad (15)$$

Definiendo:

q: Número de pruebas

V'_{ij}: Centroide $j - ésimo$ (de la prueba $i - ésima$) para ser comparado.

V_{avgj}: Posición promedio de los $j - ésimos$ centroides para el algoritmo base

$\|.\|$: Distancia euclidiana del vector

[31] T. C. HAVENS, et al. "Fuzzy c-Means algorithms for very large data," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, 2012.
[32] J. K. PARKER, L. O. HALL, AND J. C. BEZDEK, "Comparison of Scalable Fuzzy Clustering Methods," vol. 1, no. 5, PP. 10–15, 2012.
[33] Y. GU AND D. B. GOLDGOF, "Evaluating Scalable Fuzzy Clustering," pp. 873–880.

Si el porcentaje obtenido por un algoritmo con respecto al algoritmo de referencia es cercano a cero, se dice que existe una alta fidelidad, es decir entre menor sea el porcentaje *VDQ* las particiones de cada algoritmo serán más similares.

3.3.2. Cambios de pertenencia del cluster (CC%). El cambio de pertenencia del *cluster* es una métrica de comparación de fidelidad de algoritmos de *clustering* para comparar los *clusters* de los datos entre dos particiones. Por medio de un indicador de variable (δ_i) se identifica si las asignaciones de *cluster* son las mismas en ambas particiones o diferentes, asignando un valor de 0 o 1 respectivamente. Está definida como porcentaje en la siguiente ecuación[34] [35].

$$CC\% = \frac{\sum_{i=i}^{n} \delta_i}{n} * 100 \tag{16}$$

Definiendo

n: Número total de datos

δ_i: Indicador de variable el cual mide el número de datos y cuya clasificación cambia.

3.3.3. Diferencia en la calidad de la función objetivo ($DQ\ R_m$). El análisis de la diferencia en la calidad de la función objetivo J_m (4) fue reformada puesto que matemáticamente equivale al criterio (R_m)[36].

$$R_m(V) = \sum_{k=1}^{n} \left(\sum_{i=1}^{c} Dik(x_k, v_i)^{\frac{1}{(1-m)}} \right)^{1-m} \tag{17}$$

[34] J. K. PARKER, L. O. HALL, AND J. C. BEZDEK, "Comparison of Scalable Fuzzy Clustering Methods," vol. 1, no. 5, pp. 10–15, 2012
[35] Y. GU AND D. B. GOLDGOF, "Evaluating Scalable Fuzzy Clustering," pp. 873–880.
[36] Ibid

Esta nueva expresión $R_m(V)$ tiene la ventaja de no utilizar la matriz U y se puede calcular inicialmente de los centroides de los *clusters* finales. Esto quiere decir que el valor de $R_m(V)$ es más favorable que el de J_m, ya que solo requiere del conjunto de datos originales. Se usa para comparar la calidad de los resultados del algoritmo *FCM* con respecto a sus variantes algorítmicas[37].

Para medir el porcentaje de diferencia de calidad del algoritmo *FCM* con respecto a los algoritmos de aceleración se utiliza la siguiente formula:

$$DQR_m\% = \left(\frac{R_{m2} - R_{m1}}{R_{m1}}\right) * 100 \qquad (18)$$

Donde:

R_{m1}: Criterio de optimización reformulado para el algoritmo de referencia *FCM*

R_{m2}: Criterio de optimización para el algoritmo de aceleración a comparar.

[37] J. K. PARKER, L. O. HALL, AND J. C. BEZDEK, "Comparison of Scalable Fuzzy Clustering Methods," vol. 1, no. 5, pp. 10–15, 2012

4. EXPERIMENTOS

En este trabajo de grado se usó como base un algoritmo solicitado al autor *Jonathon K. Parker*[38]. El código fue revisado, ajustado y compilado bajo la herramienta GCC de Linux, para manejar datos en extensiones *.bin* o *.csv* tomando valores flotantes o enteros cortos respectivamente. Para los resultados obtenidos en los experimentos, se utilizó un computador marca Samsung NP300E4C el cual cuenta con un procesador Intel Core i5-3210M CPU @ 2.5 GHz 4, memoria RAM 4 GiB y sistema operativo Ubuntu 14.04 LTS 64 bits.

Las pruebas se realizaron con los parámetros fijados en la Tabla 3, representando como eps al criterio de parada de los algoritmos, $fPDA$ como el tamaño fraccional de la muestra inicial y $\#Ensayos$ como el número de pruebas realizadas a los algoritmos para obtener un promedio de iteraciones que realiza cada algoritmo.

Tabla 3. Asignación de valores a parámetros utilizados por los algoritmos.

Parámetro	*Valor asignado*
$fPDA$	0,2
eps	0,001
m	2
$\# Ensayos$	15

4.1. DATASETS

Se seleccionaron los siguientes *Dataset*s con tamaños y dimensiones diferentes, con el fin de hacer una comparación más amplia de las métricas cómputo de cada uno de los algoritmos estudiados.

[38] J. K. PARKER, “Jonathon K. Parker.” [Online]. Available: https://www.semanticscholar.org/author/Jonathon-K-Parker/2182375.

Los *Dataset* usados para el análisis de los algoritmos se encuentran en la siguiente página: https://archive.ics.uci.edu/ml/index.html, la cual actualmente cuenta con más de 350 conjuntos de datos como un servicio a la comunidad para el aprendizaje automático[39].

Tabla 4. Información de los *Datasets*.

Datasets	*Iris*	*Parkinsons Telemonitoring*	*3D Road Network*
Número de instancias (n)	150	5.875	434.874
Características de los atributos	Real	Entero, Real	Real
Características del conjunto de datos	Multivariante	Multivariante	Textos secuenciales
Número de atributos (d)	4	26	4

Las fuentes de cada *Dataset* se mencionan a continuación:

I. *Fuente Iris:* R.A. Fisher, Michael Marshall (Marshall% PLU '@' io.arc.nasa.gov)[40]

II. *Fuente Parkinsons Telemonitoring:* Creada por Athanasios Tsanas (tsanasthanasis '@' gmail.com) y Max Little (littlem '@' physics.ox.ac.uk) de la Universidad de Oxford, en colaboración con 10 centros médicos de los EE.UU. e Intel Corporation[41].

[39] M. LICHMAN, "UCI Machine Learning repositorio," 2013. [Online]. Available: https://archive.ics.uci.edu.
[40] M. R.A. Fisher, "The use of multiple measurements in taxonomic problems," 2005. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Iris.
[41] L. R. TSANAS, MA LITTLE, PE MCSHARRY, "Accurate telemonitoring of Parkinson's disease progression by non-invasive speech tests," 2009.

III. *Fuente 3D Road Network:* Manohar Kaul, Departamento de Ciencias de la Computación de la Universidad de Aarhus, Dinamarca (mkaul '@'cs.au.dk)[42]

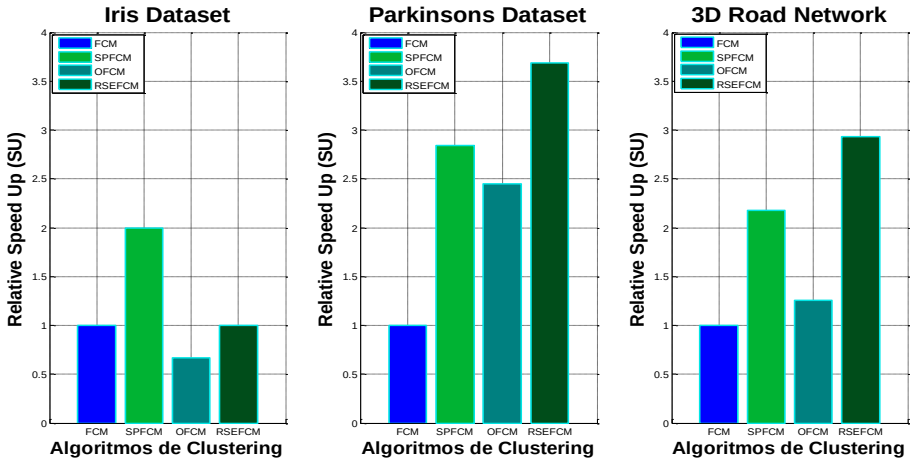
a) Resultados obtenidos del tiempo de cómputo tablas y figuras:

Se aprecia la proporción que existe entre el tiempo de ejecución de un algoritmo y con respecto a otro, indicando cuantas veces más rápido es un algoritmo.

Tabla 5. Comparación del tiempo de cómputo $SU_{1,2}$

$SU_{1,2}$	*Iris Dataset*	*Parkinsons T. Dataset*	*3D Road Network Dataset*
FCM	1	1	1
SPFCM	2	2,8439	2,1763
OFCM	0,6667	2,4496	1,2541
RSEFCM	1	3,6899	2,935

Ilustración 6. Comparación Speed Up *FCM* vs Algoritmos según cada *Dataset*



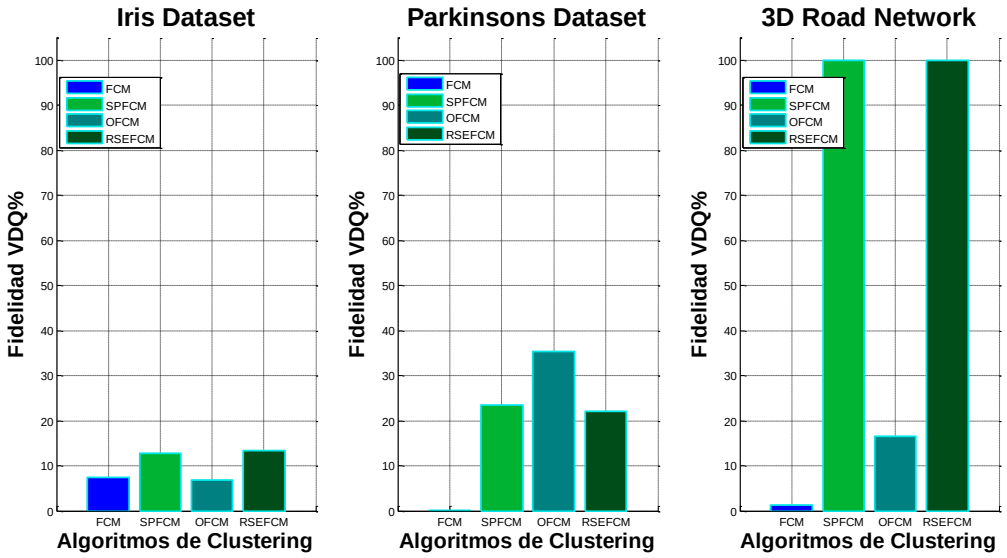
[42] M. K. CHENJUAN GUO, YU MA, BIN YANG, CHRISTIAN S. JENSEN, "Building Accurate 3D Spatial Networks to Enable Next Generation Intelligent Transportation Systems," in *IEEE*, 2013.

b) Resultados obtenidos para *VDQ*% tablas y figuras:

Tabla 6. Comparación de la calidad de las particiones *VDQ*%

VDQ%	*Iris Dataset*	*Parkinsons T. Dataset*	*3D Road Network Dataset*
FCM	7,3683	0,098	1,3181
SPFCM	12,8705	23,4882	100,0001
OFCM	6,7931	35,4047	16,5475
RSEFCM	13,4493	22,0227	100,0001

Ilustración 7. Comparación calidad de particiones *VDQ*% *FCM* vs Algoritmos respecto a cada *Dataset*



En cada figura de la Ilustración 7 se observa el cambio de la métrica *VDQ*% con respecto a cada *Dataset*. Entre mayor es el número de datos, menor es la fidelidad que existe entre los algoritmos *SPFCM*, *OFCM*, *RSEFCM* con respecto al algoritmo de referencia *FCM*. Es decir, el centroide escogido por los algoritmos es

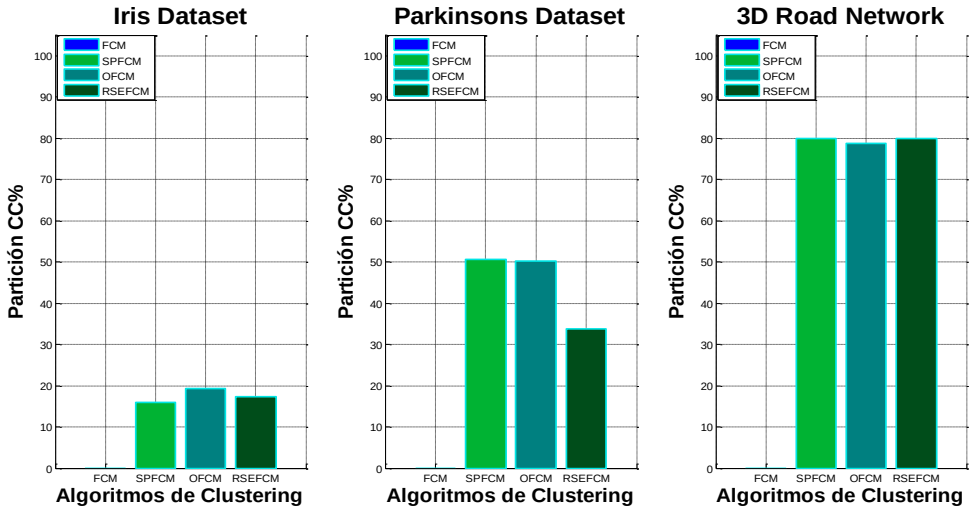
diferente en comparación al escogido por *FCM*. Se presenta una variación grande con respecto al *Dataset* 3D Road Network, donde es posible concluir que el centroide escogido por los algoritmos *SPFCM* y *RSEFCM* es completamente diferente al *FCM* en un 100%, mientras que *OFCM* mantiene una fidelidad teniendo una diferencia de centroides del 16,5%.

c) Resultados obtenidos para *CC%*:

Tabla 7. Comparación de los cambios de pertenencia *cluster CC%*

CC%	*Iris Dataset*	*Parkinsons T. Dataset*	*3D Road Network Dataset*
FCM	0	0	0
SPFCM	16	50,6723	79,8914
OFCM	19,3333	50,1787	78,7023
RSEFCM	17,3333	33,7702	79,8914

Ilustración 8. Comparación cambios de pertenencia de *cluster CC% FCM* vs Algoritmos respecto a cada *Dataset*



Observamos en la Ilustración 8 que los cambios de pertenencia de un *cluster* aumentan según el número de datos analizados. Las particiones creadas por cada algoritmo son en gran porcentaje diferentes a las que crea el algoritmo de referencia. En este caso los valores de *CC%* son altos, por lo tanto se logra identificar que la creación de particiones es ciertamente diferente para *SPFCM*, *OFCM* y *RSEFCM* respecto a *FCM*.

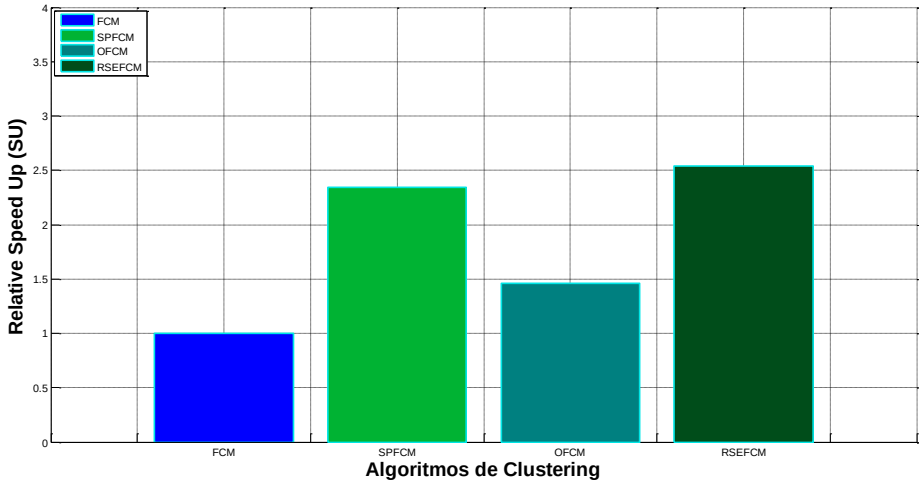
Al analizar cada *Dataset* se hace necesario extraer un promedio del speed up de cada algoritmo y de esta forma determinar cual de todos maneja un mejor tiempo de ejecución a la hora de examinar un conjunto de datos. En la Tabla 8 se observa el SU promedio.

Tabla 8. $SU_{1,2}$ Promedio FCM vs Algoritmos de *clustering*

	$SU_{1,2}$ Promedio
FCM	1
SPFCM	2,3401
OFCM	1,4568
RSEFCM	2,5416

Se puede observar en la Ilustración 9 que el algoritmo *RSEFCM* demuestra un $SU_{1,2}$ mayor con respecto al resto de algoritmos, siendo la version mas rapida con respecto a *FCM.*

Ilustración 9. Comportamiento $SU_{1,2}$ de los algoritmos de *clustering*



El algoritmo *SPFCM* trata de mantener la misma relación de proporcionalidad $SU_{1,2}$ con respecto a *FCM*, en cuanto a *OFCM* presenta un $SU_{1,2}$

Para obtener una idea más clara del comportamiento de la relación proporcional $SU_{1,2}$ se decide realizar pruebas variando el parámetro *fPDA*, utilizando para la ejecución el *Dataset* 3 ya que posee un número de datos n amplio y una dimensionalidad no tan compleja. La Tabla 9 muestra los resultados obtenidos de SU como resultado de asignar diferentes valores para el parámetro *fPDA.*

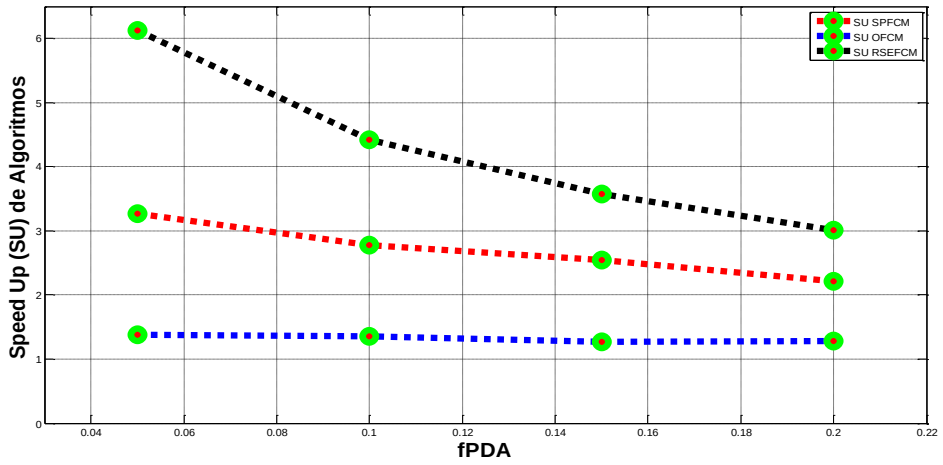
Tabla 9. $SU_{1,2}$ de *SPFCM*, *OFCM* y *RSEFCM* con respecto a *FCM*

fPDA	0,05	0,1	0,15	0,2
SPFCM	3,2741	2,7786	2,5505	2,2147
OFCM	1,3780	1,3529	1,2676	1,2850
RSEFCM	6,1301	4,4170	3,5748	3,0097

De una mejor manera es posible deducir en la ilustración 10 que el algoritmo con menor tiempo de ejecución es *RSEFCM*, puesto que presenta un $SU_{1,2} = 6{,}1301$

veces mayor con respecto a *FCM*, esto indica que el algoritmo es 6,1301 veces más rápido cuando el parámetro *fPDA* equivale a 0,05 y disminuye según va aumentado el parámetro.

Ilustración 10. Comportamiento $SU_{1,2}$ con respecto a la variación de *fPDA*



De igual manera los algoritmos *SPFCM* y *OFCM* presentan un comportamiento similar a *RSEFCM*, pero con la diferencia de que poseen un tiempo de ejecución mayor a este. Aun así siguen siendo más rápidos que el algoritmo *FCM*.

De la misma manera se realizaron pruebas para observar los cambios que presentaban las métricas *VDQ%* y *CC%* obteniendo los siguientes resultados:

Tabla 10. *VDQ%* de *SPFCM*, *OFCM* y *RSEFCM* con respecto a *FCM*

fPDA	0,05	0,1	0,15	0,2
SPFCM	100,001	100,0001	100,0002	100,0001
OFCM	14,8262	15,3321	16,464	16,5475
RSEFCM	100,001	100,0001	100,0002	100,0001

Ilustración 11. Comportamiento *VDQ%* con respecto a la variación de *fPDA*

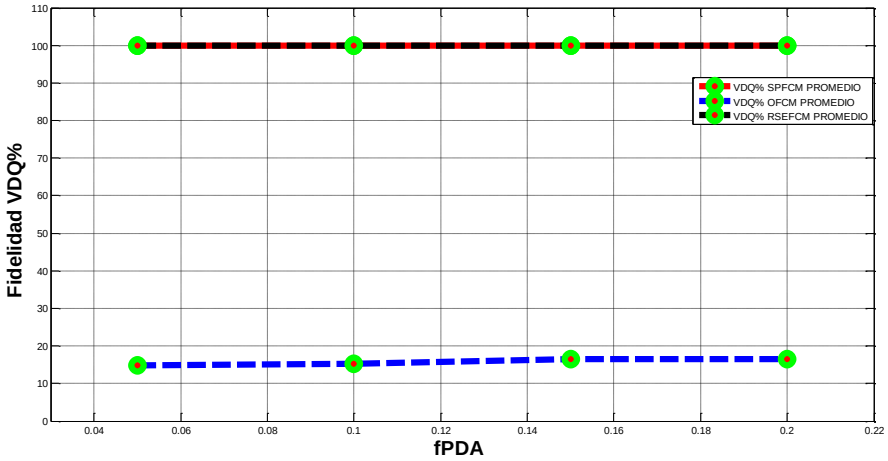
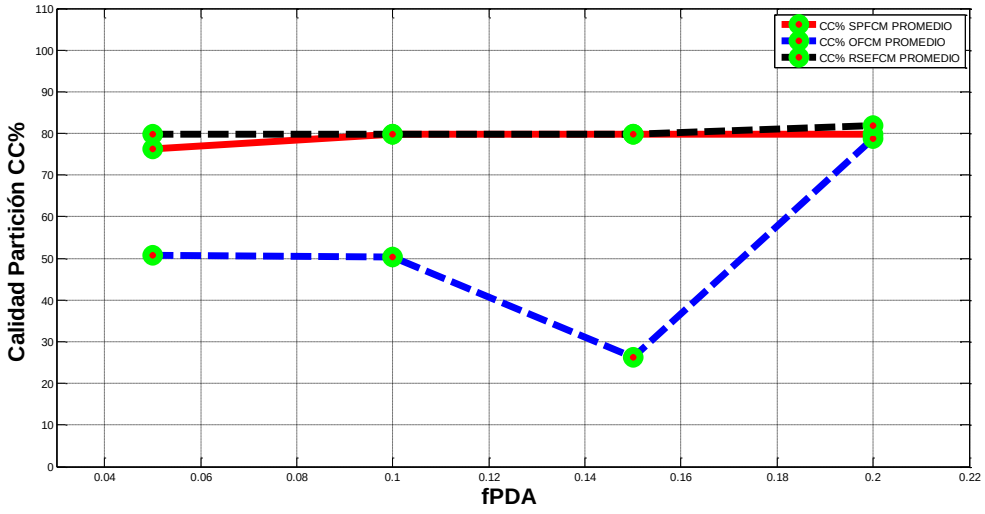


Tabla 11. *CC%* de *SPFCM*, *OFCM* y *RSEFCM* con respecto a *FCM*

fPDA	0,05	0,1	0,15	0,2
SPFCM	76,1949	79,8914	79,8914	79,8914
OFCM	50,6905	50,3077	26,2114	78,7023
RSEFCM	79,8914	79,8914	79,8914	81,9023

Ilustración 12. Comportamiento *CC%* con respecto a la variación de *fPDA*



En las Ilustraciones 11 y 12 se identifica que los algoritmos *SPFCM* y *RSEFCM* no poseen variaciones grandes en cuanto a *VDQ%* y *CC%. OFCM* presenta variaciones *CC*% cuando *fPDA* alcanza los valores de 0,15 y 0,2. Los valores de *VDQ%* y *CC%* son mayores en los algoritmos *SPFCM* y *RSEFCM*, mientras que *OFCM* mantiene un porcentaje en diferencia de calidad menor.

Para una mejor interpretación de los resultados de *VDQ% y CC%* se utilizó como métrica final la diferencia de calidad en la minimización de la función objetivo $J_m(U,V)$ expresado como *DQ* $R_m\%$ como fue definida anteriormente. La tabla 13 evidencia los valores *DQ* $R_m\%$ calculados para cada *Dataset.*

Tabla 12. Comparación *DQ* $R_m\%$ de *FCM* vs Algoritmos de Aceleración para cada *Dataset*

DQ $R_m\%$	*Iris Dataset*	*Parkinsons T. Dataset*	*3D Road Network Dataset*
SPFCM	21,6433	-0,4876	-100
OFCM	-7,1781	0,5651	-0,6445
RSEFCM	30,2266	-0,4570	-100

Al analizar los valores obtenidos por *DQ* $R_m\%$ se aprecian que existen valores negativos según cada *Dataset*, indicando que se obtuvo una minimización menor con respecto al algoritmo *FCM.* Esto quiere decir que la diferencia de calidad en la minimización de la función objetivo fue mejor que la esperada. Los algoritmos *SPFCM* y *RSEFCM,* presentan una mejor calidad con respecto al *FCM* tradicional en los *Datasets* de Parkinsons y 3D Road Network, puesto que el valor minimizado de la función objetivo es menor.

Lo anterior nos permite deducir que las métricas *VDQ%* y *CC%* no expresan la perdida de fidelidad para los algoritmos que poseen un valor de *DQ* $R_m\%$ negativo, ya que el valor R_m de un algoritmo de aceleración es un valor menor más deseable que el algoritmo de referencia *FCM*. De otra forma los valores *VDQ%* y

CC% determinan en este caso la mejora de la calidad de estos algoritmos cuyos valores fueron negativos.

4.2. TENDENCIA DE LA COMPLEJIDAD COMPUTACIONAL

Para realizar este análisis se tomó como referencia la complejidad computacional de los algoritmos descrita por el autor[43], cuyo resultado esta descrito en la Tabla 1. Para corroborar este resultado con la investigación de este libro se utilizó el *Dataset* Parkinsons Telemonitoring como referencia para realizar el estudio de la complejidad contrastada con la del autor[44].

Inicialmente se procede a encontrar un número de iteraciones y *clusters* promedios para cada algoritmo y únicamente variar el número de muestras del *Dataset*.

Los valores fijos que se tomaron para la creación de la Tabla 13 fueron $t_1 = 73{,}95$; $t_2 = 95{,}73$; $t_3 = 228{,}38$; $t_4 = 73{,}68$; $d = 26$; $s = 101$ y $c = 10$. Donde t es el número de iteraciones de los algoritmos 1, 2, 3 y 4, d el número de dimensiones del *Dataset,* c el número de *clusters* y s el número de subconjutos de datos de X definidos anteriormente.

Luego se realizaron pruebas donde al variar el número de muestras variaba el número de iteraciones, construyendo así la Tabla 14 manteniendo los valores de c,d y s anteriormente nombrados.

Tabla 13. Complejidad teórica del tiempo

[43] J. C. M. ROJAS DIAZ, Jerónimo; CHAVARRO PORRAS and R. LAVERDE, "Técnicas De Lógica Difusa Aplicadas a La Minería De Datos," *Sci. Tech.*, vol. XIV, no. x, pp. 1–6, 2008.
[44] Ibid

Muestras n	1175	2350	3525	4700	5875
FCM	225917250	451834500	677751750	903669000	1129586250
SPFCM	292447512,5	584895025	877342537,5	1169790050	1462237563
OFCM	697688680	1395377360	2093066040	2790754720	3488443400
RSEFCM	2228622,5	4457245	6685867,5	8914490	11143112,5

Tabla 14. Complejidad experimental del tiempo

Muestras n	1175	2350	3525	4700	5875
FCM	187218037,5	474747000	627527550	914453150	1336027875
SPFCM	260652600	464864075	1054135388	1235044850	1545066250
OFCM	857034425	1266694650	2067142838	2597147150	3298636250
RSEFCM	1755793.19	4320405.19	7911164.48	8802029.70	11947772.2

Ilustración 13. Complejidad del tiempo *FCM*

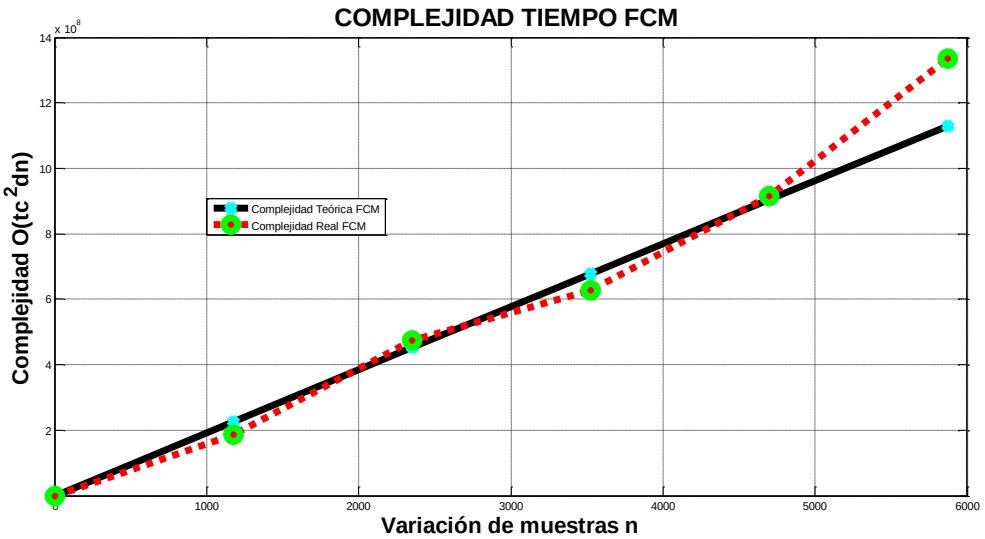


Ilustración 14. Complejidad del tiempo *SPFCM*

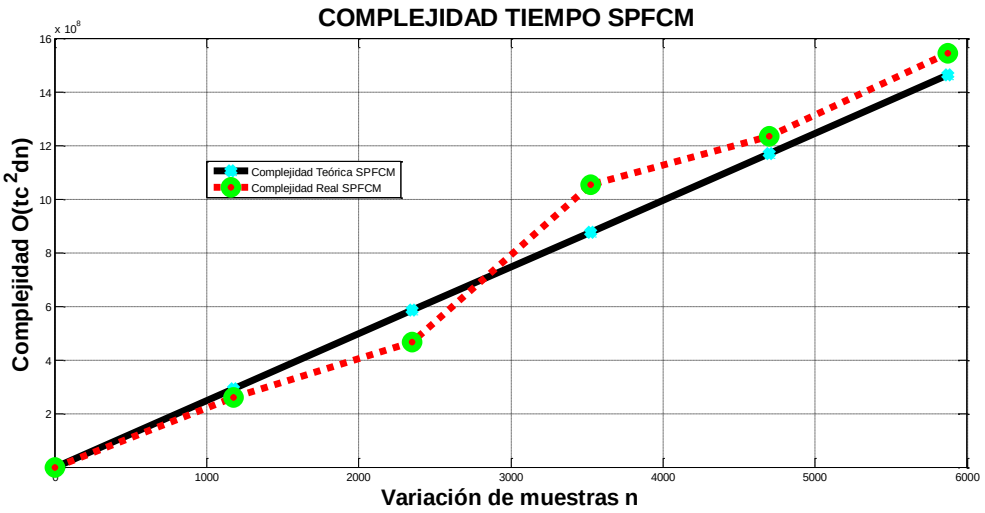


Ilustración 15. Complejidad tiempo *OFCM*

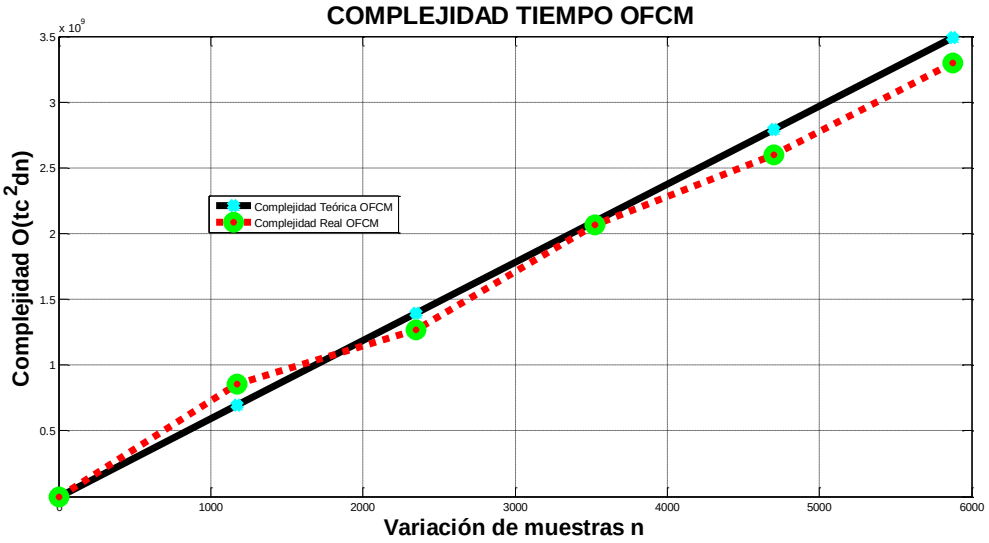
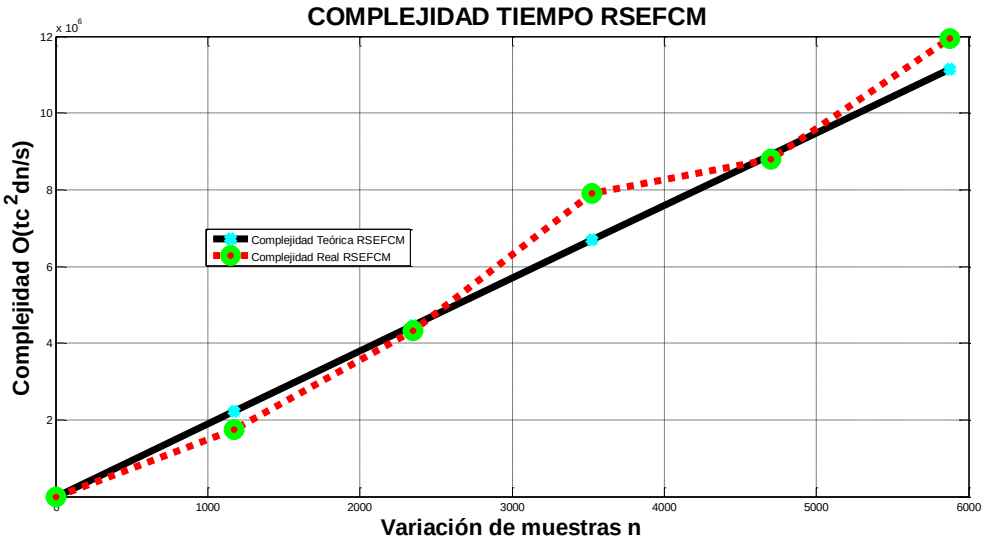


Ilustración 16. Complejidad del tiempo *RSEFCM*



El resultado del analisis tanto teorico como experimental demostro que la complejidad la complejidad computacional es de forma lineal de orden $O(n)$, la cual es una complejidad buena y tambien se usa mucho a la hora de analisis de algoritmos, en los datos de las tablas 14 y 15 se puede apreciar que la variante de *RSEFCM* arrojo valores menores con respecto a los demas algoritmos, esto quiere decir que a mayor eficiencia computacional mayor consumo de memoria y viceversa.

5. CONCLUSIONES

Con base en los desarrollos y resultados obtenidos y presentados en el presente proyecto de grado, se concluye que:

- El algoritmo con un menor tiempo de ejecución es la versión *RSEFCM*, seguida por *SPFCM* y *OFCM*, determinando que *RSEFCM* es el algoritmo que posee mayor velocidad con respecto al *FCM* tradicional. Este patrón se logra identificar por medio de la variación del parámetro *fPDA (fractional Partial Data Accesses)* identificado como la causa principal de aceleración de cada algoritmo que lo utiliza.

- Al ejecutar cada algoritmo fue posible analizar que la versión *OFCM* realiza un número de iteraciones mayor que el resto de algoritmos. La cantidad de iteraciones de los algoritmos varía con respecto a un criterio de parada previamente definido, lo que implica que al realizar un mayor número de iteraciones se tiene menos posibilidades de conseguir un mejor rendimiento en cuanto a tiempo de cómputo, como se logró observar en *OFCM* con respecto a *FCM* (Ver Ilustración 10).

- Se puede concluir que la comparación de la calidad de las particiones *VDQ*% vs algoritmos de *clustering* evaluada en cada uno de los de los algoritmos varía según las métricas de estudio, dando como resultado el porcentaje de cada métrica según el *Dataset* utilizado para las pruebas.

- Al comparar cada versión de *FCM* se logra identificar un grado de fidelidad muy bajo con respecto a *FCM*, puesto que los valores de *CC%* y *VDQ*% superan el 50% en el *Dataset* 3D Road Network. Lo anterior es posible debido a que el *Dataset* posee datos con valores grandes que dificultan el proceso de

agrupación de los mismos. Entre cada versión fue posible identificar que los algoritmos que hacen un proceso de partición y de selección de centroides totalmente diferentes son *SPFCM* y *RSEFCM.* Aunque realizan un tiempo de cómputo menor a *FCM,* los centroides y sus *clusters* son diferentes.

- Al evaluar las métricas de comparación de cambios de pertenencia de los *clusters* (*CC%*) y la calidad de particiones de los centroides (*VDQ*%) se encontraron resultados muy variables. Cada *Dataset* presenta unas métricas de calidad diferentes donde se analiza que cada partición y cada selección de centroides se realiza de manera totalmente diferente al algoritmo base. Aunque las métricas pueden dar resultados no deseables, pueden ser engañosas, ya que una diferencia considerablemente en la calidad de las particiones se podría interpretar como una mejora del algoritmo con respecto al algoritmo base.

- El estudio de la métrica *DQ* R_m*%,* demuestra la calidad que tiene un algoritmo de aceleración con respecto al algoritmo *FCM* tradicional en la minimización de la función objetivo. Demostrando que es necesario tener en cuenta el análisis de esta métrica ya que de ella dependen los valores *VDQ%* y *CC%.* Lo anterior hace referencia a que es posible encontrar un valor de *DQ* R_m*%* negativo indicando que el valor R_m calculado por un algoritmo de aceleración es menor en comparación al encontrado por *FCM* y desarrollando una mejor calidad en la minimización de $J_m(U,V)$.

- *VDQ%* y *CC%* asumen otro significado con respecto a los valores negativos encontrados en la métrica *DQ* R_m*%.* Definiendo las anteriores métricas como una mejora en la calidad de las particiones realizadas por los algoritmo de aceleración según la calidad de minimización de R_m. En este caso SPFCM y RSEFCM obtuvieron una mejora en la calidad considerablemente alta en comparación al *FCM* tradicional en el *Dataset* 3D Road Network.

- Finalmente se pudo concluir que la complejidad computacional se comporta de forma lineal obteniendo un orden $O(tc^2dn)$, de igual manera que la estudiada por el autor del código usado para el estudio[45]. Este orden lineal define que la velocidad o el tiempo de ejecución de un algoritmo es proporcional al número de muestras n como se demostró en la Ilustraciones 13, 14, 15, 16. Para ello fue indispensable realizar un promedio de la complejidad extraída del algoritmo en el *Dataset*.

[45] J. K. PARKER, "Jonathon K. PARKER." [Online]. Available: https://www.semanticscholar.org/author/Jonathon-K-Parker/2182375.

BIBLIOGRAFÍA

A. VILLAGRA and G. LEGUIZAMÓN, “Metaheurísticas aplicadas a Clustering,” universidad nacional de san Luis, 2009.

C. P. LOPEZ, Minería de datos: técnicas y herramientas, Segunda ed. España, 2008.

J. A. MAÑAS, “Análisis de Algoritmos: Complejidad,” 1997. [Online]. Available: http://www.lab.dit.upm.es/~lprg/material/apuntes/o/.

J. C. M. ROJAS DIAZ, Jerónimo; CHAVARRO PORRAS and R. LAVERDE, “Técnicas De Lógica Difusa Aplicadas a La Minería De Datos,” Sci. Tech., vol. XIV, no. x, pp. 1–6, 2008.

J. K. Parker and L. O. Hall, “Accelerating fuzzy-c means using an estimated subsample size,” IEEE Trans. Fuzzy Syst., vol. 22, no. 5, pp. 1229–1244, 2014.

J.PARKER.” [Online]. Available: https://www.semanticscholar.org/author/Jonathon-K-Parker/2182375.

J. K. PARKER, L. O. HALL, and J. C. BEZDEK, “Comparison of Scalable Fuzzy Clustering Methods,” vol. 1, no. 5, pp. 10–15, 2012.

J. W. MARÍN, A ,BRANCH B, “Aplicación de dos nuevos algoritmos para agrupar resultados de búsquedas en sistemas de catálogos públicos en línea (OPAC).” Rev. Interam. Bibl., vol. 31, pp. 47–65, 2008.

J. WU, Advances in K-means Clustering, vol. 53, no. 9. China, 2013.

L. R. TSANAS, MA Little, PE McSharry, “Accurate telemonitoring of Parkinson’s disease progression by non-invasive speech tests,” 2009.

M. GALLARDO, “Aplicación De técnicas De Clustering Para La Mejora Del Aprendizaje,” 2009.

M. K. Chenjuan Guo, Yu Ma, Bin Yang, Christian S. Jensen, “Building Accurate 3D Spatial Networks to Enable Next Generation Intelligent Transportation Systems,” in IEEE, 2013.

M. Lichman, “UCI Machine Learning repositorio,” 2013. [Online]. Available: https://archive.ics.uci.edu.

M. R.A. Fisher, “The use of multiple measurements in taxonomic problems,” 2005. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Iris.

Métodos Jerárquicos de Análisis Cluster,” 2014.
http://www.ugr.es/~gallardo/pdf/cluster-3.pdf

P. Hore, L. O. Hall, D. B. Goldgof, and W. Cheng, “Online fuzzy C means,” Annu. Conf. North Am. Fuzzy Inf. Process. Soc. - NAFIPS, pp. 1–5, 2008.

PEDRO LARRANAGA, INAKI INZA, ABDELMALIK MOUJAHID, “Clustering,” pp. 1–11. http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/t14clustering.pdf

P. LEMEY, M. SALEMI, AND A. VAMDAMME, The phylogenetic handbook, SECOND EDI. USA: 2009, 2009.

P. RAGHAVAN Christopher D. Manning, “Group-average agglomerative clustering,” 2009. [Online]. Available: http://nlp.stanford.edu/IR-book/html/htmledition/group-average-agglomerative-clustering-1.html.

S. BECA COFRE, “Clustering difuso con selección de atributos,” p. 100, 2007.

S. GHOSH and S. K. S. DUBEY, “Comparative analysis of k-means and fuzzy c-means algorithms,” Ijacsa, vol. 4, no. 4, pp. 35–38, 2013.

T. C. HAVENS, et al. “Fuzzy C-Means algorithms for very large data,” IEEE Trans. Fuzzy Syst., vol. 20, no. 6, pp. 1130–1146, 2012.

TIEMPO DE EJECUCION. Notaciones para la Eficiencia de los Algoritmos,” 19AD.

Y. GU AND D. B. GOLDGOF, “Evaluating Scalable Fuzzy Clustering,” pp. 873–880.