UN MODELO PARA LA EXTRACCION DE CONOCIMIENTO EN BASE DE DATOS MEDIANTE COMPUTO EVOLUTIVO AUTOMATAS FINITOS DETERMINISTASY REGLAS DIFUSAS.

AIDYS ARCON PINEDA

UNIVERSIDAD SIMON BOLIVAR

FACULTAD DE INGENIERIAS

MAESTRIA EN INGENIERIA DE SISTEMAS Y COMPUTACIÓN

BARRANQUILLA

2019

UN MODELO PARA LA EXTRACCIÓN DE CONOCIMIENTO EN BASE DE DATOS MEDIANTE CÓMPUTO EVOLUTIVO, AUTÓMATAS FINITOS DETERMINISTAS Y REGLAS DIFUSAS

AIDYS ARCON PINEDA

TESIS DE GRADO

Presentado como requisito de grado para optar al título de Magister en Ingeniería de Sistemas y Computación

Director
M.Sc Ing. Jonathan Ruiz Rangel

Co-Director
M.Sc Ing. Karina Jiménez Vega

UNIVERSIDAD SIMON BOLIVAR

FACULTAD DE INGENIERÍAS

MAESTRÍA EN INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

BARRANQUILLA

2019

Agradecimientos

A Dios, Gloria in excelsis Deo.

A mis padres Cástulo Arcón y Clara Pineda, por su apoyo.

A mis hermanos, por entender mi ausencia y apoyarme en cada instante, y a enfrentar con perseverancia cada reto.

A mis asesores Jonathan Ruiz Rangel y Karina Jiménez Vega, por su acompañamiento, por brindarme su conocimiento y orientación.

A mis amigos, Néstor Torres y Luis Maradei, Rafael García, Inés D`vera por infundirme fuerzas y motivarme a seguir adelante con este proyecto.

Gracias a todos

1. CONTENIDO

1. Contenido II
2. Índice de tablas. V
3. Índice de figuras VI
Resumen 7
Abstract 9
1. Introducción 11
1.1. Planteamiento del problema 12
1.2. Revisión de bibliografía 14
1.3. Objetivos 23
1.3.1. Objetivo general. 23
1.3.2. Objetivos específicos. 23
1.4. Hipótesis 23
1.5. Metodología 24
1.6. Método de investigación. 24
1.7. Desarrollo de la investigación 25
1.8. Organización de la tesis. 27
1.9. Resultados y productos esperados. 28
1.10. Alcances. 29
1.11. Aportes y contribuciones. 29
2. Capítulo 1: Marco teórico 29
2.1. La minería de datos y proceso de extracción de conocimiento en base de datos 31
2.1.1. Descubrimiento de conocimiento en base de datos. 32
2.1.2. Minería de datos. 37
2.2. Autómatas 53
2.2.1. Autómata finito determinista. 53
2.3. Optimización 56
2.3.1. Concepto de optimización. 57
2.3.2. Optimización mono-objetivo. 60
2.3.3. Optimización multi-objetivo. 62
2.3.4. Teoría combinatoria. 68
2.3.5. Optimización combinatoria. 70
2.3.6. Optimización multi-objetivo y autómatas. 71
2.3.7. Autómata finito determinista multi-objetivo. 75
2.4. Computación flexible 83
2.4.1. Algoritmos evolutivos. 87
2.4.2. Algoritmos genéticos (AG). 90
2.4.3. Metaheurística evolutiva con autómatas. 105
2.4.4. Algoritmos evolutivos con autómatas. 110

2.5. Lógica difusa 113
2.5.1. Sistema de clasificación basados en reglas difusas. 117
2.5.2. Sistema de clasificación basada en reglas difusas evolutivos. 125
2.5.3. Sistemas evolutivos difusos en minería de datos. 129
2.5.4. El problema de la alta dimensionalidad en el aprendizaje de Sistemas Basados en Reglas Difusas. 133
3. Capítulo III 138
3.1. Un modelo para extracción de conocimiento en grandes bases de datos mediante cómputo evolutivo, autómatas deterministas y reglas difusas 138
3.2. Aspectos generales de la investigación 139
3.3. Modelo EKCEAD para la extracción de conocimiento en bases de datos a través del cómputo evolutivo, autómatas deterministas finitos y reglas difusas para problemas de alta dimensión 144
3.3.1. Definición de la gramática libre de contexto. 157
3.3.2. Evaluación de la calidad de la solución. 158
3.3.3. Manteniendo la diversidad en la población: Competición elitista. 159
3.3.4. Implementación de los operadores genéticos. 160
3.3.5. Etapa de reproducción: Selección, aplicación de los operadores genéticos y reemplazamiento de la población. 161
3.3.6. Descripción del algoritmo. 164
3.4 Análisis de Complejidad de EKCEAD 167
4. Capítulo IV 173
4.1. Estudio experimental 173
4.2. Hipótesis 173
4.2.1. Variables de investigación. 173
4.2.2. Tipos de hipótesis a formular 173
4.3. Tipo de investigación. 174
4.4. Metodología. 174
4.5. Diseño experimental. 174
4.5.1. Estudio de los datos disponibles. 175
4.5.1.1. Conjunto de datos. 175
4.5.1.2. Escogencia de los criterios de comparación entre las predicciones test no paramétricos. 176
4.5.1.3. Selección de las técnicas de predicción. 176
4.5.1.4. Análisis comparativo con otros métodos de aprendizaje. 178
4.5.2. Análisis de precisión 178
4.5.3. Análisis de la comprensibilidad 181
5. Resultados obtenidos y discusión 182
6. Conclusiones 188
7. Limitaciones teórico prácticas de EKCEAD 191
8. Líneas de investigación futuras 191
BIBLIOGRAFIA 194
9. Anexos 1 206

10. Apéndices 208

2. **Índice de tablas.**

Tabla 1. Clasificación de los algoritmos colonia de hormigas 16
Tabla 2. Principales avances en los algoritmos según la naturaleza. 22
Tabla 3. Función de transición δ..... 56
Tabla 4. Función δ para el AFD I con vector de entrada $\bar{X}$=(1,2,3)..... 73
Tabla 5. Evaluación de los vectores en la función objetivo..... 74
Tabla 6. Función δ para q_0 del AFDM del ejemplo 1..... 80
Tabla 7. Función F(X)aplicada a los estados qi, qi∈Q, del AFDM del ejemplo 1..... 82
Tabla 8. Matriz de Transición EKCEAD.....157
Tabla 9.Iteraciones EKCEAD.....176
Tabla 10. Resultados de precisión de EKCEAD y otros métodos de aprendizaje de SCBRDs.....178
Tabla 11.Estadisticos y valores críticos para test de Friedman Confianza (precisión), α = 0.05.....179
Tabla 122.Ranking de Friedman 180
Tabla A.134.Propiedades de la base de dato el repositorio.....208
Tabla A145. Resumen de la experimentación de EKCEAD inicio (0 ejecución) 209
Tabla A156. Resumen de la experimentación de EKCEAD para 10 ejecuciones 211
Tabla A167. Resumen de la experimentación de EKCEAD para 20 ejecuciones 212
Tabla A178. Resumen de la experimentación de EKCEAD para 30 ejecuciones 213
Tabla A189. Resumen de la experimentación de EKCEAD para 40 ejecuciones 215
Tabla A19. Resumen de la experimentación de EKCEAD ejecución final 217

3. **Índice de figuras**

Figura 1. Fases del proceso de extracción de conocimiento en base de datos 35
Figura 2. Representación diagrama de transición del autómata finito determinista mediante el uso de diagramas de transición del AFD 56
Figura 3. Conjuntos soluciones para un problema mono-objetivo 62
Figura 4. Conjunto de soluciones mono-objetivo 64
Figura 5. Conjunto de soluciones multi-objetivo 64
Figura 6. FP Bi-Objetivo 65
Figura 7. Estructura de un estado de un AFD I 72
Figura 8. Representación del AFDI para el vector de entrada $\bar{X} = (1,2,3)$ 74
Figura 9. Estructura de un estado $q \in Q$ 75
Figura 10. Representación del espacio de soluciones factibles 76
Figura 11. AFDM construido a partir del estado inicial q0 80
Figura 12. Conjunto de soluciones factibles para el problema del ejemplo 1 83
Figura 13. Hibridación Soft computing 84
Figura 14. Pseudocódigo algoritmo evolutivo 89
Figura 15. Clasificación de los algoritmos 92
Figura 16. Pseudocódigo de un algoritmo genético 93
Figura 17. Operador de cruce 99
Figura 18. Ejemplo de cruce uniforme 99
Figura 19. Mutación que altera el valor de los cromosomas en serie 101
Figura 20. Permutación aplicada a un cromosoma 102
Figura 21. Clasificación de las metaheurística 107
Figura 22. Ciclo de algoritmo evolutivo 108
Figura 23. Diseño de un SCBRD 118
Figura 24. Ejemplo de partición difusa 119
Figura 25. Definición de tres funciones trapezoidales de pertenencia mediante cuatro genes 131
Figura 26. Entrenamiento autómata finito determinista 156
Figura 27. Codificación de una regla 158
Figura 28. Marco general mediante EKCEAD 164
Figura 29. Cruce de alelos 166
Figura 30. Cruce de alelos 167
Figura 31. Rankings de Friedman (Precisión) 179
Figura 32. Rankings de Friedman (Confianza) 180
Figura 34. Matriz de correlación EKCEAD 185

Resumen

El objetivo de esta tesis es el desarrollo de un método para la extracción de conocimiento en grandes bases de datos. Hoy en día se genera a través de la tecnología grandes cantidades de datos, tanto en volumen como en la dimensionalidad de sus variables. En este sentido, es necesario resaltar que la manipulación de los datos con un número elevado de variables presenta un problema para las técnicas tradicionales. Por otra parte, el conjunto de soluciones alternativas es tan elevado que la obtención de un óptimo global es inalcanzable en un tiempo razonable. Por tanto, es imprescindible emplear técnicas basadas en meta heurísticas que han demostrado ser una alternativa en la solución de problemas de gran complejidad por su aplicabilidad y eficiencia.

El presente trabajo propone un nuevo modelo EKCEAD (A model for the extraction of knowledge in Databases through evolutionary computation, finite deterministic automata and fuzzy rules), para la extracción de conocimiento en grandes bases de datos en forma de reglas difusas mediante el desarrollo de varias herramientas orientadas a procesos de búsqueda y optimización que utiliza las ventajas de autómatas finitos en la obtención de la mejor solución. El cual se prueba con casos reales comparándolo con otros métodos similares referenciados en la literatura, los resultados obtenidos han sido validados mediante el uso de pruebas estadísticas no paramétricas, que muestran un buen desempeño en términos de precisión e interpretabilidad.

Los resultados reflejan la bondad del modelo propuesto permitiendo que sea recomendado en la extracción de conocimiento en base de datos e incentivando su uso en futuras investigaciones.

Palabras claves: Algoritmo evolutivo, Autómatas finitos deterministas, Lógica difusa, Extracción de conocimiento, Base de reglas difusas, Optimización.

Abstract

The objective of this thesis is the development of a method for extracting knowledge in large databases. Today, large amounts of data are generated through technology, both in volume and in the dimensionality of their variables. In this sense, it should be emphasized that the manipulation of data with a large number of variables presents a problem for traditional techniques. On the other hand, the set of alternative solutions is high that obtaining an overall optimal is unattainable in a reasonable time. Therefore, it is essential to employ heuristic meta-based techniques that have proven to be an alternative in solving problems of great complexity because of their applicability and efficiency.

This work proposes a new model EKCEAD (A model for the extraction of knowledge in Databases through evolutionary computation, finite deterministic automata and fuzzy rules), for knowledge extraction in large databases in the form of fuzzy rules by developing various tools oriented to search and optimization processes that uses the advantages of finite automata in obtaining the best solution. Which is tested against real-world cases against other similar methods referenced in the literature, the results obtained have been validated using nonparametric statistical tests, which show good performance in terms of accuracy and interpretability?

The results reflect the goodness of the proposed model by allowing it to be recommended in the extraction of knowledge in database and incentivizing its use in future research.

Keywords: Evolutionary algorithm, Deterministic finite automatons, Fuzzy logic, Knowledge extraction, Fuzzy rule base, Optimization.

1. **Introducción**

En la actualidad los sistemas de información se sujetan a la estrategia de las tres V, la primera de ellas hace referencia a Volumen, entendido como la cantidad de datos que el sistema produce / consume, la segunda de ellas significa Variedad, entendida como la diversidad de los datos en relación a sus tipos y orígenes, la última V se refiere a la Velocidad a la que se generan o asimilan; existe una posibilidad de una cuarta V, que se relaciona con la veracidad de los datos. Las exigencias impuestas por estas Vs a los sistemas de procesamiento de datos obliga a desarrollar algoritmos que permitan extraer conocimiento útil y brinde soporte a la toma de decisiones; se ha recorrido un largo camino y en se han producido aciertos y desaciertos, pero no todo está dicho y pese a los avances que han tenido las técnicas para la extracción y procesamiento de datos, aún subsisten problemáticas sobre la eficiencia y efectividad de ellas, muchos de los métodos implementados con buen nivel de precisión resultan poco adaptables, es decir soluciones específicas para un problema particular, con un nivel elevado de complejidad y un alto coste computacional.

Los hechos expuestos en el párrafo anterior, incentivan la generación de algoritmos donde la eficiencia en el manejo de los datos sea un factor considerado desde su misma conceptualización, y permita diseñar nuevos métodos de optimización con mayores niveles de interpretabilidad y precisión, de alta calidad y con menor coste computacional. Hasta ahora con los métodos actuales de predicción obtienen valores aproximados que son de utilidad, pero el interés de este trabajo es mejorar la calidad de las predicciones.

Así las cosas, el presente trabajo propone un nuevo modelo metaheurístico de optimización para problemas que presentan una alta dimensionalidad, llamado EKCEAD (A model for the extraction of knowledge in Databases through evolutionary computation,

finite deterministic automata and fuzzy rules), el cual simula el comportamiento de la propagación virus ébola en las células humanas y aprovechando la inteligencia de enjambre.

Para validar el desempeño del modelo propuesto, se evaluó su comportamiento usando como problema de prueba, la generación de reglas difusas para determinar preferencias de compra, aplicados al conjunto de datos de la compañía Foodmart, se utilizan estos datos puesto que existe suficiente literatura especializada para ser utilizada como contraste. Los resultados obtenidos con el método propuesto fueron cotejados contra los obtenidos por los algoritmos A priori y FP Growth, en función de las siguientes métricas: Precisión e interpretabilidad, en base a los parámetros del costo computacional y tiempo de ejecución, es de aclarar que el comparativo se realiza bajo la premisa de generación de un conjunto de reglas de igual tamaño. Los resultados del test de Friedman muestran evidencias que apoyan la hipótesis que la propuesta es una alternativa viable para la extracción de conocimiento en grandes bases de datos.

1.1. Planteamiento del problema

En la actualidad se ha incrementado tanto la cantidad de datos obtenidos de los sistemas de información, como la necesidad de desarrollar algoritmos que permitan extraer conocimiento útil para predecir información futura. Esta inminente necesidad es propia en los Sistemas de Clasificación Basados en Reglas Difusas, a pesar de ser una herramienta muy utilizada en Minera de Datos debido a la interpretabilidad proporcionada por el concepto de etiqueta lingüística González (2007), Berlanga (2010), Kumar, Kavitha, & Ahn, (2016), Tsakiridis, Theocharis, & Zalidis, (2016) su eficiencia se ve comprometida al generar más cantidad de reglas, especialmente cuando los conjuntos de datos son grandes, esta problemática despierta especial interés, debido a que muchos de los métodos implementados con buen nivel de precisión se vuelven poco adaptables, bien por la cantidad

de reglas que utilizan, o porque su eficiencia se ve comprometida. Estas limitaciones hacen que el proceso de aprendizaje se vuelva más complicado, presentándose un elevado nivel de complejidad y un alto coste computacional por lo que requiere bastante tiempo y memoria para llegar a la convergencia (Olivas, Valdez, Melin, Sombra, & Castillo, 2019), por lo tanto ha sido necesario el diseño de nuevos enfoques evolutivos para mejorar la calidad de predicción. No obstante, el problema de aportar mejores soluciones, de interpretabilidad y precisión, haciendo uso de un menor tiempo de cómputo a través del diseño de nuevos métodos sigue siendo un problema abierto. En este sentido la meta heurística ébola se convierte en una alternativa para resolver este tipo de problemas, encuentra rutas factibles satisfaciendo múltiples restricciones en menor tiempo.

Es por ello que el presente estudio propone la formulación de un modelo basado en lógica difusa, Autómatas Finitos y algoritmo evolutivo, que adaptados a la meta heurística ébola aporte soluciones de alta calidad para problemas de alta dimensionalidad en un tiempo de cómputo real.

De este planteamiento surge la motivación de esta investigación, la cual parte de las siguientes preguntas.

¿Cómo combinar la teoría de autómatas, el cómputo evolutivo y la lógica difusa para descubrir conocimiento de interés dentro de los datos que permita describir y representar el conocimiento de forma comprensible, mediante reglas difusas?

¿La presencia de autómata finito en el aprendizaje un SCBRDs incide en los métodos de descubrimiento de conocimiento, de modo que no se produzcan repeticiones en el proceso aprendizaje y búsqueda de soluciones optimas?

¿Cómo validar la técnica utilizada, de modo, que muestre un buen equilibrio entre interpretabilidad y precisión?

¿Cuál es el nivel de calidad de respuesta que proporciona el nuevo algoritmo meta heurístico propuesto?

¿Representan esta calidad de respuesta alguna mejoría en comparación con las metas heurísticas más referenciadas en la literatura?

¿Cuáles son los valores de los parámetros que optimizan el funcionamiento del algoritmo propuesto?

1.2. Revisión de bibliografía

El proceso completo de extraer conocimiento a partir de base de datos, ha sido un tópico de investigación de creciente interés en las últimas décadas, debido a que comprende diversas etapas desde la obtención de los datos hasta la aplicación del conocimiento adquirido en la toma de decisiones. Entre esas etapas, se encuentra la minería de datos, es la más importante para el proceso de extracción de conocimiento. Esta es vital para la obtención de los resultados apropiados, y depende en gran parte del algoritmo de aprendizaje que se utilice.

En los últimos años se han realizado muchas propuestas de modificaciones a los algoritmos básicos de Minería de Datos que buscan mejorar el desempeño de estos cuando se enfrentan a problemas de alta dimensionalidad. En este contexto, en la minera de datos se realizan investigaciones, y continuamente se desarrollan algoritmos genéticos, que bajo limitaciones aceptables de eficiencia computacional descubran patrones a partir de los datos. El espacio de búsqueda de patrones suele ser muy extenso, y la extracción de patrones demanda realizar algún

tipo de búsqueda sobre el espacio. Actualmente, existen limitaciones tanto computacionales como físicas, por lo tanto, es necesario restringir este espacio en límites concretos.

Existen múltiples ejemplos de optimización combinatoria, entre los que se encuentra la extracción de conocimiento, que puede ser modelada de esta forma. El modelamiento de optimización combinatoria, generalmente se caracteriza por un conjunto finito de soluciones admisibles, el cual implica explorar dos campos fundamentales: La representación del espacio de soluciones factibles y las metaheurística para la optimización multi-objetivo.

Numerosos algoritmos meta heurísticos se han desarrollado como lo afirma Merrikh-Bayat, (2015)la naturaleza ha sido la fuente de inspiración para el diseño de metaheurística de optimización para resolver problemas complejos, como respuesta a la necesidad de encontrar alguna solución. De hecho, los métodos disponibles en la actualidad para el modelamiento de problemas de optimización combinatoria, son deudores de las investigaciones que se han realizado durante décadas pasadas. Véase la tabla 1 para una guía de los principales avances producidos.

Muchos esfuerzos se han dedicado para el desarrollo de nuevos métodos y técnicas metaheurística, en cambio pocos han sido los esfuerzos en representar el espacio de soluciones factibles a explorar. Unas de las investigaciones referente al tema fue realizada por Chaiyaratana, (1997) representó el espacio de soluciones factibles de problemas combinatorios utilizando vectores, cada vector representa una codificación (cromosoma), posteriormente Dorigo (1996) propuso un algoritmo colonia de hormigas (ACO) basado en el comportamiento natural de las hormigas para descubrir fuentes de alimentación, al establecer el camino más corto entre ellas y el hormiguero , y transmitir esta información al resto de sus compañeras. Otros modelos basados en ACO pueden encontrarse en la tabla 1.

Tabla 1.
Clasificación de los algoritmos colonia de hormigas

Investigadores	Algoritmo	Descripción
Dorigo (1996)	ACO	Ruta más corta.
Eschenauer ,(1988)	algoritmo Bicriterio (BCA)	Ruteo de vehículos
Morales & Mariano, (1997)	MOAQ	Redes de distribución del agua
Gambardella, Taillard, & Agazzi (1999)	MACS-VRPTW	Enrutamiento de vehículos de criterio doble
Doerner, Hartl & Reimann, (2001)	CA- Competants	Problemas bi-objetivos
Iredi, Merkle & Middendorf (2001)	ACO Bi-Criterion	Problemas multi-objetivos
Cardoso, Jesus.&Marquez,2003	MONA	Multi-objetive Network ACO

Fuente: Recopilación propia.

Otros modelos que representa el espacio de solución factible lo podemos ver en Kennedy.& Eberhart. R, (2001) Particle Swarrm optimization (PSO) inspirada en el comportamiento social de las bandas de pájaros o bancos de peces represento el espacio de soluciones guiadas por la partícula que mejor solución encuentre y que hace de líder en la bandada; SPEA2(Kim, Hiroyasu, Miki, & Watanabe, 2010)*Strength Pareto Evolutionary Algorithm 2* y NSGAII(Horn, Nafpliotis, & Goldberg, 2002) *Niched Pareto Genetic Algorithm* permiten la optimización de problemas combinatorios mediante técnicas de elitismo con el objeto de converger rápidamente a soluciones óptimas.

Por otro lado, no podemos dejar de mencionar las técnicas clásicas de estrategia de búsqueda local que aún siguen siendo utilizadas para la optimización multi -objetivo. Una de ellas es AMOSA Bandyopadhyay, Saha, Maulik, & Deb, (2008), ("*Multi-objective simulated annealing*") evitan en la medida de lo posible óptimos locales. Otros modelos basados en AMOSA tales como UMOSA (Ulungu, Teghem, Fortemps, & Tuyttens, 1999), CMOSA (Ulungu et al., 1999), EMOSA (Li & Landa-Silva, 2008).

Uno de los métodos más recientes en la literatura lo encontramos en Mendoza. M, (2016) ESSA un algoritmo bio-inspirado en un sistema depredador y una manada de presas para realizar una búsqueda eficiente en una región de solución factible.

Aun cuando, existen diversas técnicas para enfrentar las limitaciones encontradas en los problemas de análisis combinatorios multi-objetivos predominan las falencias en los tiempos de respuesta. Estudios más recientes para resolver problemas de optimización introducen el concepto de autómatas finitos, entre los estudios revisados los autores Niño & Ardila.(2009)proponen un Autómata Finito Determinista de Intercambio (AFD-I) al que denominaron MIDA la cual ha demostrado ser útil en la representación del espacio de soluciones factibles para un problema combinatorio. No obstante, está limitada a solucionar problemas combinatorios mono-objetivo, lo cual no permite resolver problemas más complejos.

Ahora bien, existen muchos problemas de naturaleza combinatoria que poseen más de una variable objetivo. En consecuencia, los autores Niño, Ardila, Molinares, Barrios, & Yesid(2011), proponen una metaheurística denominada MODS, inspirados en MIDA, utilizaron en su estructura un criterio de búsqueda elitista, el cual permite explorar todos los espacios de soluciones factibles, la aplicación de dicho modelo se restringe al hecho reducir el espacio de soluciones factibles a un espacio de soluciones óptima; Nieto(2011) reconoce el problema en la representación del modelo en el procedimiento propuesto por Niño et al.,(2011) en el tiempo de respuesta, al aumentar el número de objetivos y restricciones a optimizar aumenta la complejidad de encontrar el conjunto de soluciones óptimas, por lo que, plantea una ruta alternativa de solución denominada AERSMIDA la estrategia planteada evita caer en óptimos locales al utilizar dos técnicas permite una diversificación de la población aplicando el operador genético de recombinación de la población (cruzamiento) y la técnica de recocido simulado.

Un estudio posterior en el año 2011 Niño, Nieto& Chinchilla, proponen metaheurística que llamaron **EMSA** para la optimización de problemas multi-objetivo exponen una dirección de búsqueda dada a través de la asignación de pesos para cada función a optimizar del problema combinatorio; En una nueva investigación Niño, (2012**),** plantea una nueva metaheurística

EMODS basada en Autómatas Finitos Determinístico (AFDM) para la optimización multi-objetivo explora las diferentes regiones del espacio de soluciones factibles y la búsqueda de soluciones no dominadas mediante búsqueda tabú.

En ese mismo año Niño, propone metaheurística **SAMODS & SAGAMODS** plantea una dirección de Búsqueda y mejora el ángulo removiendo las soluciones dominadas cuando encuentra soluciones no dominadas asignándole pesos a los objetivos del problema para evitar caer en óptimos locales**.**

A pesar que las metaheurística enunciadas permiten dar solución a problemas combinatorios multi-objetivo prevalecen las limitaciones en los tiempos de respuestas, dado el comportamiento de las variables de salida con respecto a los parámetros del algoritmo, siendo un punto clave en la consecución de las soluciones óptimas. Ahora bien, establecer los valores a los parámetros de un algoritmo exige un análisis más profundo para conocer el comportamiento general del problema.

Las diferentes metaheurística vistas, han sido utilizadas para resolver una amplia gama de problemas combinatorios en diferentes áreas de procesos industriales, tales como: Asignación de recursos a procesos de producción (Guerra Diego, 2015), selección de la ruta optima en el problema del agente viajero TSP (Hincapie, Rios, & Gallego, 2004), el problema del binpacking, (Pérez-Ortega et al., 2016), enrutamiento de vehículos de múltiples depósitos con costos relacionados con el peso (Linfati, Escobar, & Gatica, 2014). En Rocha,; González. y Orjuela (2011) presentan una revisión bibliográfica de las diferentes variantes surgidas dentro Problema de Ruteo de Vehículos, tales como enrutamiento de vehículos de múltiples depósitos, problemas de enrutamiento del vehículo de entrega y recogida con ventanas de tiempo, el problema de vendedores ambulantes ferroviarios, problema del vendedor ambulante de forma mono o multi objetivo, entre otros.

Como puede apreciarse son muchos los problemas de optimización combinatoria presentes en la realidad, y su modelación se define de acuerdo a la técnica que se vaya a adaptar. Tal es el estudio realizado por Ruiz, (2013)diseñó e implementación de la metaheurística **ERNEAD** basada en Autómatas Finitos Deterministas Multi-Objetivo con algoritmos genéticos (EMODS) permite dar solución al problema de entrenamientos de redes neuronales con cálculo hacia adelante (feedforward) asignando pesos sinápticos a las neuronas, aplicó Framework a problemas de diagnóstico del cáncer de mama en el área de la medicina y clasificación de flores en el área botica, procedentes de la base de datos tomadas del MLR.

Sea otro caso de estudio de optimización realizado por Hernández Riaño, López Pereira, & Hernández Riaño,(2013a) proponen un algoritmo denominado Ébola basado en el comportamiento del virus para resolver el problema de enrutamiento en redes UnicastQoS, cuya intención es encontrar una ruta óptima que permita ofrecer un mejor servicio de comunicación, que dependiendo de algún criterio de desempeño "minimizar el retraso" es visto como un problema de optimización para determinar la ruta más corta a lo largo de la ruta seleccionada.

Es importante señalar que los métodos de aprendizaje de los SCBRD se pueden abordar de una mejor manera con técnicas de optimización. Entre los métodos de aprendizaje están las técnicas estocásticas y difusas: Las Redes neuronales (J. Rua, 2014); computación flexible: métodos evolutivos y genéticos (Vaezpour, Dehghan, & Yousefi'zadeh, 2019), (Naghash Asadi, Abdollahi Azgomi, & Entezari-Maleki, 2019); lógica difusas, clasificadores bayesianos (Naudin, Tremblais, Guillevin, Guillevin, & Fernandez, 2002); Técnicas bayesianas: Algoritmo EM (Simon, Weber, & Evsukoff, 2008); Técnicas basadas en árbol de decisión y sistemas de aprendizaje de reglas: el algoritmo C4.5 Y el algoritmo ID3 (J.R, Quinlan, 1993)entre otros.

En consecuencia, el aprendizaje de reglas de clasificación es un problema clásico del aprendizaje automático, la construcción de clasificadores es una de las técnicas utilizadas

comúnmente en la minería de datos (Motoda et al., 2007). Entre los algoritmos clásicos está el ID3, y el C4.5 los cuales inducen arboles de decisión, (Y. F. Cabrera, 2011) propone un método para construcción de relaciones de similaridad, a partir de la cual se pueda mejorar el desempeño de métodos de descubrimiento de conocimiento; Un estudio realizado por Hasperue (2013) plantea un método denominado CLUHR utilizando híper-rectángulos para extraer conocimiento en forma de reglas de clasificación.

Dado el gran volumen de datos de los problemas con el que los algoritmos de aprendizaje de reglas de asociación trabajan, y la tarea de buscar patrones que cumplan con ciertos requisitos, implica, que sea viable la construcción de clasificadores con sistemas difusos, su implementación ha sido mayor en problemas de descubrimiento de conocimiento. El uso de algoritmos evolutivos para el descubrimiento de reglas de asociación difusas ha dado lugar a los Sistemas basados en reglas difusas evolutivos (SBRDEs), un trabajo realizado por Gonzalez, (2007), implemento un algoritmo evolutivo para la extracción de reglas difusas para la tarea de descubrimiento de subgrupos, denominado SDIGA, cuyo objetivo es la generación de reglas individuales que aporten conocimiento interesante y encontrar relaciones desconocidas entre las variables las cuales pueden ser continuas o categóricas. véase (Del jesus, González, & Herrera;Francisco, 2005), (Del Jesus, González, & Herrera, 2007), (DelJesus, Herrera, Mesonero, & Gonzalez, 2008), (Carmona, 2011).Otro estudio realizado por González (2007) MESDIF (Multiobjetive evolutionary subgroup discovery fuzzy rules) sostiene que a pesar del número elevado de variables y los valores perdidos, bajo número de ejemplos y pocas variables continuas el algoritmo multi-objetivo, permite obtener conjuntos de reglas fáciles de interpretar, con un nivel alto de confianza, de soporte y de completitud.

Posteriormente Berlanga (2010) propone GP-COACH desarrollo un algoritmo de aprendizaje de SCBRDs compactos y precisos, que muestre un buen equilibrio entre la interpretabilidad y la

precisión en problemas que presentan una alta dimensionalidad, para ello hace uso de los algoritmos evolutivos (AEs), particularmente la programación genética y lógica difusa; Otro estudio realizado por el mismo autor Berlanga (2010) GP-CO^{2}ACH (Genetic programing based coevolutionary learning of compact and accurate fuzzy rule based classification system for high dimensional problems) para ello utiliza un algoritmo y dos especies distintas, una de ellas aprende la BR reglas (GP-CO^{2}ACH-P) y la otra aprende la DB (GP-CO^{2}ACH-S), evolucionan en paralelo o secuencialmente, cooperando entre sí para formar una solución completa a un problema dado.

Martinez. R, (2014) presentó el diseño de una herramienta para procesar conjunto de datos que contienen datos de baja calidad, comprende dos paquetes, el primero NIPip trabaja con un conjunto de datos cada vez y el segundo ExpNIPip está orientado a la experimentación y procesa varios conjunto de datos simultáneamente.

Recogiendo lo más importante de los investigadores Berlanga, Niño, Nieto, y Ruiz coinciden en el análisis que el principal inconveniente está dado por el crecimiento exponencial que se produce en el espacio de búsqueda de reglas difusas, con un aumento lineal en el número de variables, lo que es denominado como el problema de la explosión combinacional de reglas, cuyo crecimiento dificulta el proceso de aprendizaje y, en la mayoría de las ocasiones conlleva a obtener una base de reglas de elevada cardinalidad, lo que disminuye el nivel de interpretabilidad del sistema. Convirtiéndose en un sistema más complicado y complejo con respecto al número de reglas, variables y etiquetas incluidas en cada regla. Lo que puede hacer que el algoritmo de aprendizaje de SCBRDs quede estancado en un óptimo local o que se produzca sobre aprendizaje, es decir que el algoritmo sea capaz de aprender un SCBRDs que cubra los ejemplos de entrenamiento, pero, que sin embargo presente una elevada tasa de error ante nuevos ejemplos de prueba.

Por otro lado, en la literatura especializada no se encontraron estudios para el descubrimiento de reglas de asociación difusa en los cuales se hayan implementado estrategias de búsqueda con autómatas finitos deterministas. Es por esto, que es desconocido su aplicabilidad en el ámbito de los sistemas difusos. Sin embargo, es necesario acentuar que los modelos expuestos han sido aplicados exitosamente en la solución de muchos problemas reales, por lo que no se pretende demeritar su utilización, sino analizar su comportamiento y efectividad en el proceso de extracción de reglas difusas.

Tabla 2.
Principales avances en los algoritmos según la naturaleza.

AUTOR	**DESARROLLO**
Rechenberg (1965)	Traslación Estable de la Royal Aircraft
Holland (1975)	Algoritmo Genético (GA)
Kirkpatrick et al.(1983)	Simulated Anneling
Glover (1989)	Búsqueda Tabú
Dorigo (1992)	Granja de Hormigas (ACO)
Eberhart en (1995)	Vuelo de las Aves Migratorias
Storn y Price (1997)	Evolución Diferencial
Abbass (2001)	Algoritmo de Colonia de Abejas Artificiales
Moscato y Cotta (2004)	Algoritmo Mimético
Baudry et al.(2005)	Algoritmo Bacteriológico
F C Yang 2007	Flujo de Agua
Yang (2008)	Algoritmo Luciérnaga
Yang (2009)	Búsqueda Cucú
Eskandar et al.(2012)	Water Cycle", o "Ciclo del Agua"
Tang et al.(2012)	Búsqueda de Lobo o Wolf Search
Hernández ,López , & Hernández,(2013)	*Ébola
Luo, Li, & Chen, (2013)	Salto de Ranas
Cheng y Prayogo (2014)	Búsqueda de Organismos Simbióticos
(Kaveh y Mahdavi (2014)	Choque de Cuerpos
Moosavian y KasaeeRoodsari (2014)	Liga de Fútbol
Salimi (2015)	Búsqueda Estocástica de fractales
Yazdani y Jolai (2015)	Algoritmo León
Kuo y Zulvia (2015)	Gradiente Evolución (GE)
Meng et al.(2015)	Algoritmo Murciélagos
Niu et al.(2015)	Mosca de la Fruta FOA
Fong et al.(2016)	Algoritmo Eidético, mejorar búsqueda Lobo

Fuente: Recopilación propia.

1.3. Objetivos

1.3.1. Objetivo general.

Desarrollar un modelo de análisis predictivo, que utilice técnicas de inteligencia artificial y exponga el verdadero valor de la información para la definición de estrategias y la toma de mejores decisiones

Para realizar este objetivo general, previamente se definen los siguientes objetivos particulares**.**

1.3.2. Objetivos específicos.

- Desarrollar un algoritmo de reglas de clasificación, de modo que el modelo permita la extracción de reglas difusas en forma normal disyuntiva (reglas DNF) en las que cada atributo que interviene en la regla puede tomar más de un valor.
- Aplicar al modelo un autómata finito (AF) que permita representar el espacio de soluciones factibles del conjunto de reglas de la base de conocimiento. Para ello, es necesario revisar la literatura existente de algoritmos evolutivos Multi-objetivos.
- Validar que el método desarrollado es efectivo en la solución de problemas de clasificación en casos reales. Para ello se aplicará el modelo a un conjunto de datos de prueba que presenten una alta dimensionalidad.

- Comparar los resultados obtenidos del modelo propuesto con otros modelos existentes en la literatura especializada.

1.4. Hipótesis

Se formularon las siguientes hipótesis de investigación:

Hi: “Los SCBRDs que utilizan autómatas finitos en su estructura obtienen mejores resultados de predicción, mostrando un buen equilibrio entre la interpretabilidad y la precisión de conocimiento extraído”.

Variables de investigación.

En el proceso de medición de los datos y del modelo propuesto, es necesario enunciar las variables objeto de estudio.

Calidad de respuesta: teniendo en cuenta las siguientes variables.

Precisión: proximidad de las soluciones con el valor optimo, para cada función de prueba que arroje el algoritmo de comparación.

Interpretabilidad: análisis e interpretación de los datos.

1.5. Metodología

El tipo de estudio a implementar en el desarrollo de este proyecto es de carácter descriptiva explicativo puesto que el objeto fundamental de investigación es descubrir conocimiento de interés dentro de los datos; intentando obtener información que describa y permita representar el conocimiento de forma comprensible, mediante reglas difusas generadas por la metaheurística propuesta durante el proceso de extracción.

1.6. Método de investigación.

El método implementado durante el desarrollo del proyecto de tesis es de carácter inductivo, ya que se parte de lo particular; como es el concepto de reducir o minimizar el número de reglas incluidas en la base de reglas, la interpretabilidad, así como el tiempo de ejecución de cada una de ellas, este concepto puede adaptarse de manera general a cualquier tipo de problemas que requieran optimizar o minimizar una base de conocimientos de alta dimensionalidad.

1.7. Desarrollo de la investigación

Para implementación del proyecto se desarrollarán cuatro etapas las cuales se describen a continuación:

a) ***Revisión bibliográfica.*** Exploración de los conceptos necesarios para el desarrollo de la investigación.

En esta etapa de la investigación se desarrollarán las siguientes actividades:

Elaboración del Marco teórico de las áreas de las ciencias involucradas como optimización, teoría de autómatas finitos, minería de datos y computación flexible, lógica difusa, así mismo el estudio de los sistemas actuales. Esto servirá de base para referenciar los avances tecnológicos de los modelos evolutivos de extracción de reglas difusas.

b) ***Selección de herramientas para el diseño del modelo.***

Se determina cuáles son las herramientas que se emplean para realizar el objeto de estudio de esta investigación, como es el algoritmo evolutivo, notación lingüista de la base de reglas, autómatas finitos.

En esta etapa de la investigación se desarrollarán las siguientes actividades:

- Investigar y documentar la gramática libre de contexto.
- Investigar y documentar el esquema de representación de reglas adecuado.
- Investigar y documentar los algoritmos de agrupación para particiones difusas.
- Investigar y documentar las medidas de calidad o función fitness que permita evaluar la bondad de cada solución.
- Investigar y documentar el uso de mecanismos que favorezca la diversidad dentro de la población de reglas.
- Investigar y documentar el tipo de operador genético que se debe utilizar en la adaptación de la metaheurística ébola.

- Investigar y documentar el esquema de remplazamiento de la población en la adaptación de la metaheurística ébola.
- Investigar y documentar el autómata finito.

c) **Desarrollo de la herramienta computacional**

En esta etapa se comienza a construir la estructura del modelo, el cual es muy importante conocer el funcionamiento, la comunicación y la interoperabilidad de los componentes.

En esta etapa de la investigación se desarrollarán las siguientes actividades:

- Crear la base de conocimiento que contiene:
  - Tanto el conjunto de las reglas difusas en notación normal disyuntiva que compone la Base de Reglas.
  - Como la función de pertenencia de las variables implicadas.
- Desarrollar el algoritmo genético evolutivo para agrupamiento difuso y/o búsqueda.
- Emplear un autómata finito determinista (AFD) que permita representar el espacio de soluciones factibles del conjunto de reglas de la base de conocimiento.

d) **Aplicación y validación del modelo.**

En esta etapa se realizan las pruebas y mediciones necesarias que permitan determinar el funcionamiento de la metaheurística desarrollada y la viabilidad de la implementación del modelo.

En esta etapa de la investigación se desarrollarán las siguientes actividades:

- Aplicar el modelo a diversos casos reales que presenten una alta dimensionalidad en los datos con el fin de probar la funcionalidad y desempeño para su posterior documentación.

- Seleccionar técnicas clásicas de extracción de reglas difusas evolutivas existentes en la literatura especializada y comparar los resultados obtenidos con los resultados del modelo propuesto.
- Seleccionar las métricas pertinentes que permitan verificar y validar los resultados obtenidos del modelo propuesto con los obtenidos por los otros algoritmos clásicos de descubrimiento de conocimiento mediante extracción de reglas difusas.
- Analizar el tiempo de ejecución del modelo propuesto de manera que se pueda verificar y validar los resultados en contraste con los resultados de otros modelos de extracción de reglas difusas.
- Realizar el diseño de experimento estadístico que permita responder a la hipótesis planteada por el trabajo del proyecto de investigación.
- Realizar el análisis de complejidad del modelo propuesto con el propósito de compararlo con los algoritmos clásicos de extracción.

La investigación se finalizará con conclusiones fundamentales sobre ella y propondrá líneas de investigación para el futuro.

1.8. Organización de la tesis.

En el Capítulo 1, se presenta una breve introducción sobre la temática de la tesis, se plantea la problemática y la necesidad de extraer conocimiento a partir de reglas difusas, finalmente se exponen el objetivo general que persigue la investigación y los objetivos específicos que desarrollan la investigación. Por último, los antecedentes relacionados con el área de investigación y el alcance.

En Capítulo 2, se introducen los conceptos de extracción de conocimiento en base de datos y minería de datos, y se describe en profundidad la tarea de inducción predictiva de clasificación. Posteriormente se describe la computación flexible, centrándonos dentro de las técnicas que la componen, en la descripción de algoritmos evolutivos y la lógica difusa. Por otra parte, se plantea los conceptos y teorías sobre autómatas finitos deterministas multi objetivo, optimización. Finalmente, se propone el desarrollo de un nuevo modelo para el problema de la convergencia prematura en el aprendizaje de SCBRD utilizando computación flexible.

En el Capítulo 3, se presenta una propuesta de algoritmo evolutivo para la extracción de reglas difusas que puedan extraer conocimiento utilizando reglas de tipo DNF. Se realizará un análisis de diversos componentes de dicha propuesta para comprobar su adecuación. Posteriormente se compararán los resultados de la propuesta con los obtenidos por otros algoritmos de extracción de reglas difusas. Finalmente se aplicará el modelo propuesto sobre problemas reales.

En Capítulo 4, se resume el trabajo realizado y los resultados obtenidos en este trabajo, se presentará las conclusiones extraídas sobre los mismos y se plantean trabajos futuros derivados de la misma.

Por último, se incluye un apéndice con la descripción de los test estadísticos utilizados en esta memoria. El proyecto termina con una recopilación bibliográfica que recoge las contribuciones más destacadas en la materia estudiada.

1.9. Resultados y productos esperados.

La propuesta evolutiva presentada en este proyecto en procesos de minería de datos, se espera de ella, no tanto extraer un conjunto de reglas completo que maximice la precisión global en la predicción sino obtener un conjunto parcial de reglas que maximicen la comprensibilidad del conocimiento extraído.

1.10. Alcances.

En esta investigación solo se abordará la extracción de conocimiento mediante reglas difusas evolutivas, en notación DNF, de modo que cada variable toma como valor un conjunto de etiquetas lingüísticas.

1.11. Aportes y contribuciones.

La extracción de conocimiento en forma de reglas que permite conocer las características de los datos de estudio y, la obtención de un modelo capaz de realizar predicciones en bases de conocimiento. El desarrollo de un método que permite adaptarse a diferentes bases de conocimiento de alta complejidad.

2. Capítulo 1: Marco teórico

Uno de los problemas a los que nos enfrentamos en las últimas décadas, es el trabajar con el creciente aumento del volumen y variedad de información que se encuentra informatizada en base de datos digitales y que hace necesaria la utilización de métodos automáticos para su tratamiento. El método tradicional de convertir los datos en conocimiento consiste en el análisis e interpretación de forma manual realizado por un especialista experto en la materia. Esta forma de actuar con los datos es lenta, cara y altamente subjetiva. De hecho, como el volumen de datos crece

exponencialmente, el análisis manual se torna impracticable. Por ello, es necesario el uso de metodologías y herramientas de extracción de conocimiento que lleven a cabo un análisis automático e inteligente de los datos (Hernandez, Ramirez, & Ferri, 2004).

El principal objetivo de la minería de datos es resolver problemas analizando los datos presentes en las bases de datos. Para ello integra numerosas técnicas de análisis de datos y extracción de modelos que permite predecir resultados y/o descubrir relaciones entre los datos. Así, en función del problema a resolver, la extracción de conocimiento se puede abordar desde dos perspectivas distintas desde el punto de vista predictivo, en el que se intenta pronosticar el comportamiento del modelo, o desde el punto de vista descriptivo, donde se intenta descubrir patrones que describan los datos.

El descubrimiento de conocimiento en base de datos (Knowledge Discovery in Data bases) está formado por un conjunto de pasos interactivos e iterativos, entre los que se incluye el procesamiento de los datos, en este proceso se pueden corregir los posibles datos erróneos, incompletos, inconsistentes, así como la reducción del número de registros y características encontrando entre ellos los más representativos, lo cual conlleva a la búsqueda de patrones de interés particular y a la interpretación de estos patrones. Este proceso es el más característico del KDD porque incluye el uso de algoritmos específicos para la extracción de patrones a partir de los datos, el cual es conocido como minería de datos (data-mining).

En el ámbito de competencia que caracteriza al siglo XXI, la minería de datos es usada por las compañías con el fin de generar ventajas competitivas. Dentro de este contexto es de vital importancia examinar los datos almacenados en los sistemas de información, sin embargo, esto labor presenta inconvenientes al momento de elegir la herramienta computacional por su alto coste, o por las limitaciones de acceso de las herramientas GNU, por lo que han hecho surgir nuevas técnicas metaheurística capaces de extraer patrones ocultos, describir tendencias, predecir

comportamientos y, en general, de sacar provecho a la información digitalizada, generalmente heterogénea y en grandes cantidades.

El proceso de extracción de conocimiento en base de datos continúa evolucionando a través de la investigación disciplinar de áreas como base de datos, aprendizaje automático, reconocimiento de patrones, estadística, la inteligencia artificial, la computación de alto rendimiento (Soft-Computing). La Computación flexible que incorpora teorías y técnicas, como Algoritmos Evolutivos (AE_s), lógica difusa, esta propuesta introduce el concepto de los autómatas finitos a la extracción de conocimiento en base de datos.

Esta propuesta se centra en aquellos procesos de la minería de datos orientados a la predicción, y específicamente en la clasificación. En los problemas de clasificación, los datos son objetos caracterizados por los atributos que pertenecen a diferentes clases establecidas y la meta es inducir un modelo para poder predecir la clase a la cual pertenece un objeto dado los valores de sus atributos (Weiss & Kulikowski, 1991).

En este capítulo se presenta los conceptos necesarios para el desarrollo de los siguientes capítulos. Comienza con un breve repaso del concepto de minería de datos, tarea de inducción predictiva, abordaremos el concepto autómatas finitos, optimización, computación flexible haciendo énfasis en los algoritmos evolutivos y la lógica difusa, por último, abordaremos los sistemas de clasificación basados en reglas difusas.

2.1. La minería de datos y proceso de extracción de conocimiento en base de datos

La minería de datos es el proceso de extraer conocimiento útil y comprensible, previamente desconocido, de grandes cantidades de datos almacenados en distintos formatos (Witten & Frank, 2000), de tal manera que se puedan encontrar modelos inteligibles a partir de los datos, el cual debe ser un proceso automático o semiautomático y el uso de los patrones descubiertos deberá ayudar

a la toma de decisiones, y a su vez generar algún beneficios a las organizaciones. Con frecuencia se emplean términos como sinónimos de la minería de datos uno de ellos análisis inteligente de los datos, otro término muy utilizado, y el más relacionado con la minería de datos es la extracción o descubrimiento de conocimiento en base de datos (Knowledge Discovery in Data bases, KDD). De hecho, ambos términos se utilizan indistintamente, a pesar que existen claras diferencias entre los dos, algunos autores utilizan el término KDD para referirse al proceso completo de descubrimiento útil a partir de los datos, mientras, la minería de datos es solo una fase, si bien es la más importante de este proceso y consiste en la aplicación de algoritmos específicos para la extracción de patrones a partir de datos. Al proceso de extracción de conocimiento KDD se incorpora diferentes técnicas ampliamente utilizadas en los campos del aprendizaje automático, la estadística, la base de datos, los sistemas de toma de decisión, la inteligencia artificial y otras áreas de la informática y de los sistemas de la gestión de información.

Las diferentes técnicas de minería de datos que serán vistas a continuación, han sido utilizadas para resolver una amplia gama de problemas de distinta naturaleza. Entre los trabajos existentes, el presentado en (Ma, Guo, Liu, Ma, & Chen, 2009) donde los autores hacen una revisión de los aportes realizados en minería de datos en diferentes áreas como climatología recursos naturales, predicción de desastres naturales

2.1.1. Descubrimiento de conocimiento en base de datos (Berlanga, 2010), (Hernandez, Ramirez, & Ferri, 2004).

El objetivo fundamental de este estudio KDD es, por tanto, que el usuario sea capaz de utilizar apropiadamente las diversas técnicas existentes en cada una de las fases de extracción de conocimiento a partir de datos: La recopilación de datos mediante almacenes de datos o de manera directa, la preparación de datos mediante la visualización, agregación, limpieza o transformación, la minería de datos mediante técnicas descriptivas o predictivas, la evaluación o mejora de modelos

mediante validación cruzada, combinación o análisis de costes y, finalmente, la difusión y uso del conocimiento extraído, mediante estándares de intercambio de conocimiento, en fin, dar a conocer y proporcionar una metodología que permita la escogencia tanto de herramientas como determinar que técnicas son más fiables, más eficientes, más comprensibles, y, en definitiva, más pertinente para un problema en cuestión. Se trata de describir los problemas a los que se enfrentan las organizaciones y particulares, la metodología para resolverlos y las técnicas necesarias.

Se puede definir el KDD como el proceso no trivial, de identificar patrones válidos, novedosos, potencialmente útiles y, en última instancia, comprensibles a partir de los datos (Fayyad, Piatetsky-shapiro, & Smyth, 1996a). En esta definición, se resume cada una de las propiedades deseables para el conocimiento extraído.

- **Los datos**: Son un conjunto de hechos (como ejemplo la base datos), y un
- **Patrón**: Es una expresión que describe un subconjunto de los datos o un modelo aplicable a este subconjunto
- **Proceso**: Que incluye distintos pasos, como la preparación de los datos, la búsqueda de patrones, la evaluación del conocimiento, y el refinamiento, todo ello repetido en múltiples iteraciones.
- **No trivial**: Que es necesaria cierta búsqueda o inferencia, es decir, no es un cálculo directo como puede ser calcular la media de un conjunto de números.
- **Válidos**: Los patrones deberán seguir siendo precisos para datos nuevos (con cierto grado de incertidumbre), y no solo para aquellos que han sido usados en su obtención.
- **Novedoso**: Que aporte algo desconocido tanto para el sistema y preferiblemente para el usuario.

- **Potencialmente útil**: La información debe conducir algún tipo de beneficio para el usuario.
- **Comprensible**: La extracción de patrones debe facilitar la toma de decisiones. Si no, inmediatamente, si después de someterse a algún post-procesamiento.

Lo anterior implica que podemos definir medidas cuantitativas para evaluar los patrones extraídos. En muchos casos, es posible definir medidas de certidumbre o utilidad (como la ganancia, por ejemplo, en dinero que se ha ahorrado, debido a las mejores predicciones o al menor tiempo de respuesta del sistema). Ciertos conceptos como novedad y comprensibilidad son subjetivos y por tanto son difíciles de plasmar en una medida cuantitativa. En ciertos contextos, la comprensibilidad se puede estimar mediante la simplicidad (por ejemplo, el número de bits necesarios para describir un patrón). También suele utilizarse el interés, definido como una medida general del valor de patrón que combina validez, novedad, utilidad, y simplicidad. Las funciones de interés se pueden definir de forma explícita o manifestarse implícitamente mediante una ordenación llevada a cabo por el sistema de KDD sobre los patrones o modelos descubiertos. Así, se puede considerar que un patrón es conocimiento si supera cierto umbral de interés, definido por el usuario y específico del dominio.

El KDD es un proceso de extracción de conocimiento que consta de un conjunto de etapas: La utilización de bases de datos junto con algún tipo de selección, limpieza, pre procesado, transformaciones y proyecciones de esta información; la aplicación de métodos de minería de datos para la extracción de los patrones; y por último la evaluación y posible interpretación de los mismos.

El proceso completo, se muestra en la Figura 1, es interactivo e iterativo, en el que el experto deberá tomar decisiones en distintas fases.

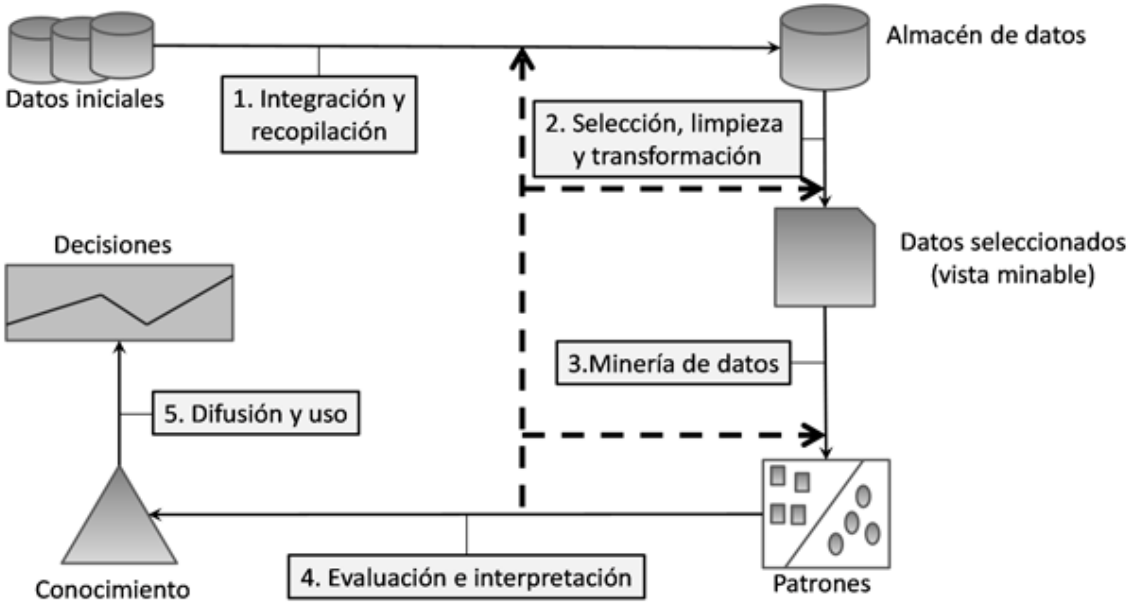


Figura 1. Fases del proceso de extracción de conocimiento en base de datos
Fuente: Adaptado de (Hasperue, 2013).

De forma general, el proceso involucra los siguientes paso (Fayyad, Piatetsky-shapiro, & Smyth, 1996b)

Integración y recopilación de los datos. En esta fase se determinan las fuentes de información que pueden ser útiles y donde conseguirlas, en esta fase se transforman todos los datos a un formato común, generalmente en un almacén datos se unifica toda la información recogida, detectando y resolviendo las inconsistencias para finalmente obtener una vista previa y decidir qué aspectos de interés pueden ser analizados.

Preparación de los datos. En esta fase se analizan y documentan los datos disponibles y las fuentes de conocimiento del dominio de la aplicación, para estudiar las características de los datos. Además, se aplicará procesamiento para mejorar la calidad de los datos disponibles para el proceso de minería, incrementando la eficiencia al reducir el tiempo de cálculo necesario. Consta de los siguientes subprocesos.

- Limpieza de datos: dado que los datos provienen de diferentes fuentes pueden contener valores erróneos o faltantes, en esta etapa se eliminan o corrigen los datos incorrectos y se decide la estrategia a seguir con los datos incompletos, para luego clasificar los datos relevantes con el objeto de realizar la tarea de minería y por consiguiente obtener los mejores resultados. Involucra varias operaciones básicas, como la normalización manejo de ruido y valores incompletos o perdidos, reducción de la redundancia entre otros.
- Integración de datos: Esta fase incluye la integración de conjunto de datos múltiples y heterogéneos generados a partir de diferentes fuentes.
- Reducción de proyección de datos: Incluye la búsqueda de características útiles de los datos según el objetivo final, de forma que se puedan evaluar y desarrollar las hipótesis y modelos iniciales, también incluye la reducción del número de variables y la proyección de los datos sobre espacios de búsqueda en los que sea más fácil encontrar una solución. Esta situación es crítica dentro del proceso que, con frecuencia, marca la diferencia entre el éxito y el fracaso en la minería de datos.

Minería de datos. En esta fase se seleccionan y aplican a los datos los algoritmos de búsqueda para encontrar patrones de interés, dicho esto, lo primero que hay que tener en cuenta es la elección del algoritmo de minería de datos a aplicar en función del objetivo del proceso del KDD (clasificación, regresión, segmentación, detección de desviaciones), y la forma de representación del conocimiento (árboles de decisión, reglas, etc.) también es necesario especificar un criterio de evaluación que permita definir qué modelo es mejor y la estrategia de búsqueda a realizar.

Evaluación e interpretación. Esta fase se encarga de medir la calidad de los modelos aprendidos, así como de realizar nuevos experimentos regresando a los pasos anteriores si es necesario, en ocasiones puede, que esta regresión, quizás utilice otros datos, otros algoritmos, otras metas y estrategias. La interpretación necesita de otras técnicas para la visualización de modelos que permitan interpretar a los usuarios finales el conocimiento que aporta los modelos aprendidos.

Difusión y uso. Tiene como fin el empleo de forma correcta del modelo aprendido en el contexto de la aplicación real y de los usuarios para los cuales se inició el proceso de extracción de conocimiento.

En el proceso de KDD es iterativo ya que puede necesitar varias iteraciones (regresar a pasos anteriores) en cualquiera de sus fases para extraer conocimiento de alta calidad, sin embargo, la fase que más complejidad demanda es la de minería de datos, esto no demerita la operatividad de las demás fases, todas ellas tienen igual importancia en el éxito del KDD.

2.1.2. Minería de datos.

Hemos visto que la minera de datos no es más que un paso esencial de un proceso más amplio cuyo objetivo es el descubrimiento de conocimiento en base de datos. El término minería de datos se usa comúnmente para denotar el hallazgo de patrones útiles en los datos. Consiste en la aplicación de análisis de datos y algoritmos de descubrimiento para producir patrones o modelos a partir de los datos pre-procesados (Fayyad et al., 1996a). Hay que resaltar que el espacio de patrones suele ser infinito, y que la extracción de patrones supone realizar algún tipo de búsqueda en el espacio, que, además, puede implicar una limitación computacional al establecer límites sobre el sub-espacio y puede ser explorado por algún algoritmo de minería de datos hasta encontrar con certeza los valores de interés.

El primer paso dentro del proceso de minería de datos consiste en determinar, qué tipo de tarea de minería es el más apropiado (ejemplo clasificación), el siguiente paso es elegir el tipo de modelo (ejemplo árbol de decisión), por ultimo elegir el algoritmo de minería que resuelva la tarea y obtenga el tipo de modelo que estamos buscando; esta elección es pertinente porque existen muchos métodos para construir los modelos. Tras asegurarnos haber realizado la elección adecuada en el tipo de tarea y modelo, continua la siguiente etapa de evaluación en la cual se refinan los datos corrigiendo o eliminando los posibles errores, y finalmente continuar con la fase de difusión y uso, expresividad y la comprensibilidad del modelo, hasta llegar a la toma de decisiones. Consecuentemente, el carácter hipotético de la minería de datos supone que lo aprendido, puede en cualquier momento ser refutado por evidencia futura, pues no es un modelo perfecto sino aproximado y siempre existirán problemas que serán resueltos mejor por otros métodos.

A continuación, se introducen los orígenes de la minería de datos, los objetivos propios del proceso de minería de datos, así como los tipos de algoritmos de minería de datos que se pueden utilizar, por último, se describen los componentes algorítmicos de la minería de datos.

2.1.2.1. Orígenes de la minería de datos.

Uno de los problemas a los que nos enfrentamos en las últimas décadas, es el trabajar con el creciente aumento del volumen y variedad de información que se encuentra informatizada en base de datos digitales. Consecuentemente, la extracción de información útil en dichos conjuntos de datos se torna una actividad compleja y llena de desafíos, debido a que las herramientas tradicionales de análisis de datos no son eficientes con conjuntos de datos de elevado tamaño, y muchas relaciones ocultas potencialmente útiles pueden no ser reconocidas.

La minería de datos, aparece como una herramienta capaz de mezclar métodos tradicionales de análisis de datos con nuevos y robustos algoritmos para hacer frente a los retos y desafíos presentados por estos grandes volúmenes de datos.

2.1.2.2. Tipos de algoritmos de minería de datos en función del objetivo.

Los objetivos de la minería de datos se definen por el uso que se pretende del sistema. Podemos distinguir dos tipos de objetivos: La verificación y el descubrimiento.

- En la verificación, el sistema se limita a verificar las hipótesis del usuario.
- En el descubrimiento, el sistema encuentra nuevos patrones de forma autónoma.

El descubrimiento se puede descomponer además en:

Predicción, donde el sistema encuentra patrones para predecir el comportamiento futuro (utiliza un conjunto de variables de la base de datos para predecir valores futuros desconocidos de otras variables de interés); y descripción, donde sistema encuentra patrones para presentarlos a un experto en una forma comprensible para él, y que describen y aportan información de interés sobre el problema y el modelo que subyace bajo los datos.

Muchos métodos de minería de datos se encuentran en la literatura hay que resaltar que realmente sólo existen unas pocas técnicas fundamentales para dicha tarea. El modelo de representación utilizado en un método suele ser una integración de técnicas; es decir, que muchos de los métodos son híbridos de varias técnicas. De esta forma, unos algoritmos difieren de otros fundamentalmente en el criterio de calidad o el método de búsqueda utilizados para evaluar el ajuste del modelo.

Como se ha mencionado, los dos objetivos en minería de datos y, en general descubrimiento de conocimiento son predicción y descripción. A continuación, se detallan los distintos tipos de tareas que se pueden realizar para ambos tipos de modelos:

Tareas que utilizan modelos predictivos citado (Hernandez, Ramirez, & Ferri, 2004)

- Clasificación. Los datos son objetos caracterizados por atributos que pertenecen a diferentes clases definidas. La meta es inducir un modelo para poder predecir la clase a que pertenece un objeto de datos dados los valores de los atributos (Hand, 1981), (Weiss & Kulikowski, 1991), (Calders & Verwer, 2010), destacan el aprendizaje supervisado mediante reglas de clasificación (Clark & Niblett, 1989), (Cohen, 2014). Estos algoritmos confían en la supervisión humana para entrenarse en la clasificación de datos en clases predefinidas de valores categóricos (nominales).
- Regresión. En esta tarea, la variable sobre la que se quiere hacer la predicción es continua. La meta es inducir un modelo para poder predecir el valor de la clase dados los valores de los atributos(J., 1992), (Breiman, Friedman, Olshen, & Stone, 1984). La regresión asume que los valores de la variable objetivo cuadran con algún tipo de función conocida (lineal, logística, etc.) y entonces determina la mejor función de este tipo que modela a los datos disponibles.
    - Análisis de series temporales. En el análisis de series temporales (Seymour, Brockwell, & Davis, 2006), (Montgomery, Jennings, & Kulahci, 2008),se examina el valor de un atributo según va cambiando con el tiempo. Los valores se suelen obtener como instantes de tiempo distribuidos de forma homogénea.

Tareas que utilizan modelos descriptivos:

- Agrupamiento (clustering). Consiste en la separación de los datos en subgrupos o clases interesantes; se busca por tanto identificar un conjunto finito de categorías o clusters que describan los datos (Alhajj & Kaya, 2008), (Street, 1982). Las clases pueden ser exhaustivas y mutuamente excluyentes o jerárquicas y con solapamientos.

- Sumarización. Son métodos para proporcionar al usuario información comprensible para captar la esencia de grandes cantidades de información almacenadas en una base de datos (Agrawal, Mannila, Srikant, Toivonen, & Verkamo, 1996). Las técnicas de sumarización suelen aplicarse al análisis de datos interactivos y a la generación automatizada de informes.
- Asociación. Es el descubrimiento de relaciones de asociación o correlaciones entre un conjunto de elementos (Alhajj & Kaya, 2008) Suelen expresarse en forma de reglas mostrando parejas atributo-valor que ocurren frecuentemente juntas en un conjunto de datos dado. En estos casos, se utiliza un modelo no supervisado de aprendizaje (Parthasarathy, Zaki, Ogihara, & Li, 1997), (Agrawal et al., 1996) cuyo objetivo es encontrar reglas individuales que definan patrones interesantes en los datos.
- Descubrimiento de subgrupos. En este dominio se lleva a cabo la búsqueda de subgrupos en el conjunto de datos que sean estadísticamente más interesantes, siendo tan grandes como sea posible y ofreciendo el mayor valor de atipicidad estadística con respecto a la propiedad en que estemos interesados (Konijn, Duivesteijn, Kowalczyk, & Knobbe, 2013), (Luc, Marc, Ali, & Hammal, 2019).

2.1.2.3. Componentes de los algoritmos de minería de datos (Berlanga, 2010).

Una vez definido el objetivo de la minería de datos, para construir algoritmos específicos que implementen los métodos generales que se ha enumerado debemos definir los tres componentes fundamentales en cualquier algoritmo de minería de datos:

- El lenguaje de representación del modelo,
- El criterio de evaluación del modelo, y
- El método de búsqueda

Esto constituye una visión simplificada (más no completa) pero bastante útil para expresar los conceptos clave de un algoritmo de minería datos de forma relativamente unificada y compacta

Lenguaje de representación del modelo. Es el lenguaje que se utiliza para describir los patrones que se descubren. Es muy importante conocer las restricciones y suposiciones que impone la representación empleada. Si la representación es demasiado limitada, no podremos producir un modelo adecuado de los datos, aunque dediquemos mucho tiempo al entrenamiento. Es importante que el analista comprenda las suposiciones sobre la representación que pueden ser inherentes a un método particular, y que el diseñador del algoritmo establezca claramente qué representación está utilizando en el algoritmo. Hay que resaltar que un gran poder de representación para los modelos incrementa el peligro de sobre ajustar los datos de entrenamiento, pudiendo obtener una precisión predictiva reducida (F. Berlanga, 2010)

Criterio de evaluación del modelo. Berlanga (2010) como la función que permite establecer hasta qué punto se ajusta bien un patrón particular (un modelo y sus parámetros) a los objetivos del proceso de KDD. Por ejemplo, los modelos predictivos suelen evaluarse por la precisión predictiva empírica sobre un conjunto de prueba, utilizando técnicas de validación cruzada (Hastie, Tibshirani, & Friedman, 2009). Los modelos descriptivos se pueden evaluar a partir de la novedad, la utilidad y la comprensibilidad del modelo. Actualmente se están utilizando con frecuencia las curvas ROC (Receiver Operating Characteristic), (Provost & Fawcett, 2001) para evaluar algoritmos**.**

Método de búsqueda. Consta de dos componentes: La búsqueda de parámetros y la búsqueda del modelo. Conocido el enfoque del lenguaje de representación y el criterio de evaluación del modelo, el problema de minería datos se traduce a una tarea de optimización: Encontrar los parámetros y modelos que optimicen el criterio de evaluación. En la búsqueda de parámetros, el algoritmo debe buscar los parámetros que optimicen el criterio de evaluación del modelo, dados

los datos y un modelo fijo de representación. Algunos de los métodos de búsqueda utilizados son la búsqueda exhaustiva, la vuelta atrás o la búsqueda probabilística.

2.1.2.4. Técnicas de evaluación (Hernandez, Ramirez, & Ferri, 2004) .

Hasta el momento hemos resaltado varias para técnicas para la generación de modelos a partir de las evidencias formadas por los conjuntos de datos, pero, como podemos saber si un determinado modelo es lo suficientemente valido, seguro y confiable en el propósito de la toma de decisiones. Esto, en algunas ocasiones puede determinarse mediante un análisis estadístico, otras veces se opta por estimar la confiabilidad del modelo. Sin embargo, es determinante conocer que existen varios métodos que sirven para evaluar la calidad de un modelo a partir de la evidencia.

1) Validación de hipótesis basada en precisión

   En los modelos predictivos, el uso de la partición del conjunto datos de entrenamiento y testeo es fácil de interpretar. Una aproximación es utilizar el propio conjunto de entrenamiento como referencia para evaluar la calidad del modelo, pero este método, es equivocado por que favorecen los modelos que sobre ajustan el conjunto de datos de entrenamiento (overfitting) y no generalizan para otros datos (sub ajuste o underfitting), consecuentemente, una mejor alternativa es evaluar los modelos sobre un conjunto de datos diferente al conjunto de entrenamiento, para ello existen diversas técnicas de evaluación basadas en partición de datos en dos partes, una para el aprendizaje (entrenamiento), y otra para la evaluación (test).

   Habitualmente, la partición se realiza de manera aleatoria, dejando el 50% de los datos para el entrenamiento, y el 50% restante ara el subconjunto de prueba. Sin embargo, puede ocurrir que dos experimentos realizados sobre la misma evidencia y con el mismo método de aprendizaje, obtengan resultados muy dispares, otro problema es

que se tienen pocos datos y reservar parte de ellos para el test puede hacer que se utilicen menos para el entrenamiento, obteniendo peores modelos.

2) Validación cruzada

El método de validación cruzada ***(****cross-validation*) consiste en dividir el conjunto de datos en *k* subconjuntos de tamaño similar, de tal modo, se reserva un grupo del conjunto de datos de prueba y con los *k*-1 restantes se emplea para construir el modelo, calcular el error de la muestra parcial y para predecir el resultado de los datos del grupo reservado. Este procedimiento se repite *k*veces, dejando siempre un subconjunto de datos para la prueba, lo cual significa que se calculan *k* tasas de error independiente. Finalmente se construye un modelo con todos los datos y se obtienen sus tasas de error y precisión promediando las *k* tasas de error disponibles. Mediante este modelo se especifican ciertas ventajas una de ellas es que la varianza de los *k* errores de muestra parciales permite estimar la variabilidad del método de aprendizaje. En general para la construcción del modelo se suele realizar 10 particiones (***10****-fold cross-validation*), otra ventaja de este método es que los *k* subconjuntos son independientes. No obstante, los conjuntos de entrenamiento no lo son por ejemplo en una validación cruzada con *k* =10 cada par de subconjunto de entrenamiento comparten el 80% de los datos. Y por lo tanto este solapamiento entre los conjuntos de entrenamiento podría afectar la calidad de la estimación, por ello Dietterich, (1998) propone una mejora en la técnica que permita utilizar subconjuntos de entrenamiento independientes, dicha mejora plantea realizar una validación cruzada 5x2 (5x2*cross-validation*) y consiste en aplicar 5 repeticiones de una validación cruzada con *k* =2. En donde cada una de las 5 iteraciones de datos se divide en dos subconjuntos disjuntos (entrenamiento y prueba) de idéntico tamaño, finalmente el error final se calcula promediando los cinco errores de las muestras parciales.

Es necesario resaltar bajo qué condiciones se puede utilizar cada método de validación cruzada, el método de validación cruzada 5x2 es conveniente utilizarla cuando las técnicas de aprendizaje son notablemente eficientes para ejecutarse 10 veces, sino lo son, se puede optar por utilizar la clásica partición de los datos en entrenamiento y prueba.

3) Bootstrap.

Es utilizada en casos los que se tienen pocos datos. El método Bootstrap, es aplicado para estimar la medida de precisión de los estimadores muéstrales denominado el error estándar del estimador. Esta técnica permite la estimación de la distribución muestra de cualquier estimador utilizando únicamente un método de re muestreo. Lo anterior consiste en construir un modelo con un conjunto de entrenamiento inicial, que posteriormente se reemplaza en función de los resultados anteriores, sin tener en cuenta que el conjunto de entrenamiento puede contener datos repetidos, o en su defecto pueden no tener algunos datos del conjunto original, estos datos que no hacen parte del muestreo se reservan para el conjunto de test, al igual que la evaluación cruzada el proceso anterior se repite un número prefijado de k veces y después se promedia los errores/precisiones (Ramírez. J, Osuna. I, Rojas. J, & Guerrero. S, 2016).

2.1.2.5. Medidas de evaluación de los modelos (Hernandez, Ramirez, & Ferri, 2004).

Dadas las características de la tarea de minería de datos existen diferentes medidas de evaluación de los modelos.

Si tarea es clasificación. Lo habitual es evaluar la calidad los patrones encontrados con respecto a su precisión predictiva. La cual se calcula así: Número de instancias del conjunto de

prueba clasificadas correctamente dividido por el conjunto de instancias totales del conjunto de prueba.

Si la tarea es asociación: Habitualmente se evalúa separada cada una de las reglas con objeto de restringir aquellas que puedan aplicarse a un mayor número de instancias y que tienen una precisión relativamente alta sobre esas instancias. Para evaluar una regla de asociación se suelen practicar dos medidas para conocer la calidad de la regla: Cobertura y Confianza.

Cobertura (denominada Soporte) de una regla se define como el número de instancias, en las que la regla se puede aplicar y predecir correctamente, es decir, el número de transacciones en las cuales los ítems en una regla ocurren juntos, en relación con el número total de transacciones.

$$Soporte\ (A \rightarrow B) = \frac{\sum x_p \epsilon\ N\mu AB(x_p)}{|N|} \quad \textbf{(1)}$$

Confianza (denominada Precisión) mide el porcentaje de veces que la regla se cumple cuando se puede aplicar.es decir, el porcentaje de transacciones que contienen conjuntamente el término del antecedente y el termino de consecuente en relación al número de transacciones que contienen la parte del antecedente,

$$Confianza\ (A \rightarrow B) = \frac{\sum x_p \epsilon\ N\mu AB(x_p)}{\sum x_p \epsilon\ N\mu A(x_p)} \quad \textbf{(2)}$$

En donde |N| es el número total de ejemplos, $\mu_A(x_p)$ es el grado de emparejamiento del ejemplo $x_p$con el antecedente de la regla y $\mu_{AB}(x_p)$ es el grado de emparejamiento del ejemplo $x_p$con la regla completa.

Una regla de asociación difusa podría ser considerada una regla de clasificación difusa si el antecedente de la regla contiene los ítems difusos (cada ítem difuso es una tupla formada por una variable y un número lingüístico) y el consecuente una clase. Así una regla de asociación difusa como $A \rightarrow Clas_j$ podría ser medida en términos de soporte y confianza como:

$$Soporte\left(A \rightarrow Clas_j\right) = \frac{\sum x_p \epsilon Clas_j \, \mu A(x_p)}{|N|} \quad \textbf{(3)}$$

$$Confianza\left(A \rightarrow Clas_j\right) = \frac{\sum x_p \epsilon \, Clas_j \, \mu A(x_p)}{\sum x_p \epsilon \, N\mu A(x_p)} \quad \textbf{(4)}$$

Si la tarea es de regresión (valor numérico), habitualmente se mide el error cuadrático medio del valor predicho respecto al valor que se utiliza como validación. Este método promedia los errores y tiene más en cuenta aquellos errores que se desvían más del valor predicho denominada ponderación cuadrática.

Si tareas es de agrupamiento las medidas de evaluación suelen depender del método utilizado, aunque suelen ser función de la cohesión de cada grupo y de la separación entre grupos. La cohesión y separación entre grupos puede formalizar, por ejemplo, utilizando la distancia media al centro del grupo de los miembros de un grupo y la distancia media entre grupos respectivamente. El concepto de distancia y densidad son dos aspectos cruciales tanto en la construcción de modelos de agrupamiento como en su evaluación.

Existen otras medidas más subjetivas para evaluar modelos como el Interés, la novedad, la simplicidad y la comprensibilidad.

2.1.2.6. Interpretación y contextualización (Hernandez, Ramirez, & Ferri, 2004).

Pese a todas las medidas vistas en el párrafo anterior, la siguiente consideración es evaluar también el contexto donde el modelo se va a utilizar. En la clasificación y las reglas de asociación, usar la precisión como medida de calidad tiene ciertas desventajas. La primera de ellas, no tiene en cuenta el problema de tener distribuciones de clases no balanceadas. Este aspecto pone en evidencia que es necesario conocer mejor el tipo de errores y su costo asociado. En los problemas

de clasificación se usa la matriz de confusión, la cual muestra el recuento de casos de las clases predichas y sus valores actuales.

La consideración de que todos los errores no son iguales puede incluso tenerse en cuenta en situaciones donde los costos de error suelen ser difíciles de estimar o incluso desconocidos para muchas aplicaciones. En estos casos, se usan estrategias alternativas como el análisis ROC (Receiver Operating Characteristic).

En efecto, por todo lo anterior, la precisión de un modelo no garantiza que refleje el mundo real. Normalmente, esta situación se produce cuando al construir el modelo no hemos tenido en cuenta algunos parámetros que implícitamente influyen en él. En cualquier caso, se deberá contrastar el conocimiento que éste nos proporciona con el conocimiento previo que pudiéramos tener sobre el problema para detectar y en su caso resolver los posibles conflictos.

2.1.2.7. Fase de difusión, uso y monitorización (Hernandez, Ramirez, & Ferri, 2004).

Una vez construido y validado el modelo puede usarse principalmente con dos finalidades:

Para que el analista recomiende acciones basándose en el modelo y en sus resultados, o bien para aplicar el modelo a diferentes conjunto de datos, también es importante incorporar el modelo a otras aplicaciones puede ser de forma manual o por medio de una herramienta de software, lo realmente indispensable es aplicar su difusión, es decir que se distribuya y se comunique a los posibles usuarios dentro de la organización, es decir, el nuevo conocimiento extraído debe integrar el *know-how* de la organización. Todo esto, es con el objeto de medir lo bien que el modelo evoluciona. Aun cuando el modelo funcione bien es necesario comprobar continuamente las prestaciones del mismo. Esto se debe principalmente a que los patrones pueden cambiar.

2.1.2.8. Arboles de decisión (Hernandez, Ramirez, & Ferri, 2004).

Uno de los sistemas de aprendizaje más utilizados son los basados en árboles de decisión, en cierto modo, es el método más fácil de utilizar y de entender. Un árbol de decisión es un conjunto de condiciones organizadas en una estructura jerárquica, de tal manera que la decisión a tomar se puede determinar siguiendo las condiciones que se cumplen desde la raíz del árbol hasta alguna de sus hojas. Esto constituye una gran ventaja dadas las opciones posibles a partir de una determinada condición son excluyentes, esto permite analizar una situación y, siguiendo el árbol de decisión apropiadamente, llegar a una sola acción o decisión a tomar.

Los árboles de decisión son especialmente adecuados para la tarea de clasificación. De hecho, clasificar es determinar de, entre varias clases, a qué clase pertenece un objeto, por tanto, asume que las clases son disjuntas, es decir, una instancia es de la clase A o de la clase B, pero no puede ser al mismo tiempo de las clases A y B. En este sentido, la clasificación se diferencia de la categorización, donde se permite más de una clase, etiqueta o categoría para cada instancia. Debido al hecho de que la clasificación trata con clases o etiquetas disjuntas, un árbol de decisión conducirá un ejemplo hasta una y sólo una hoja, asignando, por tanto, una única clase de ejemplo. Para ello, las particiones existentes en el árbol deben ser también disjuntas. Es decir, cada instancia cumple o no cumple una condición, que deben ser excluyentes y exhaustivas, Estos algoritmos se llaman algoritmos de partición o algoritmos de *divide y vencerás* (Morillo Torres, Moreno, & Díaz, 2014).

En su estructura, el algoritmo va construyendo el árbol (el contiene la raíz) añadiendo particiones y los hijos resultantes de cada partición. Lógicamente, en cada partición, los ejemplos se van dividiendo entre los hijos. Finalmente, se llega a la situación en la que todos los ejemplos que caen en los nodos inferiores son de la misma clase y esa rama ya no sigue creciendo. La única condición que hay que exigir es que las particiones al menos separen ejemplos en distintos hijos, con lo que la cardinalidad de los nodos irá disminuyendo a medida que se desciende en el árbol.

En síntesis, el método de aprendizaje de un árbol de decisión consta de:

- Particiones a considerar.
- Criterio de selección de particiones.

Particiones. Las particiones son, como se ha mencionado, un conjunto de condiciones exhaustivas y excluyentes. Lógicamente, cuantos más tipos de condiciones se permitan, habrá más posibilidades de encontrar los patrones que hay detrás de los datos y cuantas más particiones se permitan más expresivos podrán ser los árboles de decisión generados y, probablemente, más precisos. No obstante, cuantas más particiones se elijan, la complejidad del algoritmo será mayor. Por tanto, la clave en un buen algoritmo de aprendizaje de árboles de decisión es encontrar un buen compromiso entre expresividad y eficiencia.

En consecuencia, la mayoría de algoritmos de aprendizaje de árboles de decisión sólo permiten una coyuntura muy limitada de particiones. Por ejemplo, el algoritmo más popular, denominado C4.5 (Quinlan. J. R, 1993), ID3 (Quinlan. J. R, 1986), contiene un solo tipo de partición para los atributos nominales y un solo tipo de partición para los atributos numéricos. La expresividad resultante de las particiones se conoce como expresividad proporcional cuadricular. El término proporcional se refiere a que son particiones que sólo afectan a un atributo de un ejemplo a la vez, es decir, ni relacionan dos atributos del mismo ejemplo, ni dos atributos de distintos ejemplos. El término cuadricular hace referencia al tipo de particiones que realizan, especialmente cuando se refiere a los atributos numéricos. Aunque las particiones de atributos nominales y atributos numéricos resultan bastantes sencillas, permiten obtener árboles de decisión bastante precisos y muy comprensibles.

Criterio de selección de particiones. citado (Hernandez, Ramirez, & Ferri, 2004) Basándose en la idea de buscar particiones que discriminen o que consigan nodos más puros, se han presentado en las últimas décadas numerosos criterios de partición, tales como el criterio del error esperado,

el criterio Gini (Breiman. L, 1984), los criterios Gain, Gain Ratio y la modificación del C4.5 (Quinlan. J. R, 1993) y el DKM (Kearns.M. & Mansour. Y., 1995). Estos criterios de partición buscan la partición *s* con la menor I (s), definido de la siguiente forma:

$$I(s) = \sum_{j=1..n} p_j * f\left(p_j^1, p_j^2, \ldots, p_j^c\right) \quad \textbf{(5)}$$

Donde *n* es el número de los nodos hijos de la partición (número de condiciones de la partición),$p_j$esla probabilidad de caer en el nodo j, p_j^1 es la proporción de elementos de la clase 1 en el nodo j, p_j^2 es la proporción de los elementos de la clase 2 en el nodo j, y así para las c clases. Bajo esta fórmula general, cada criterio de partición implementa una función *f* distinta. Estas funciones *f* se denominan funciones de impureza y, por lo tanto, la función *I*(*s*) calcula la media ponderada (dependiendo de la cardinalidad de cada hijo) de la impureza de los hijos de una partición.

Existen muchas variantes de estos criterios por ejemplo el Gain Ratio y el criterio usado en el C4.5 son variantes de la entropía, la ortogonalidad de GID3 (Fayyad., 1994), y muchos otros que no se ajustan a la fórmula anterior por ejemplo criterios basados en el principio MDL Mínimum Description Length,(C., Hernández. J., & Ramírez. J, 2001) o criterios basados en el AUC Área underthe ROC Curve (Ferri. C., Hernández. J., & Ramírez. J, 2002). Según experimentos realizados, al parecer ser tanto el criterio usado por el C4.5 (Gain Ratio modificado) como el criterio DKM se comportan ligeramente mejor que el GINI y bastante mejor que el Error Esperado.

Poda y restructuración (Hernandez, Ramirez, & Ferri, 2004)**.** Los algoritmos para el aprendizaje de árboles de decisión ofrecen un modelo que es completo y consistente con respecto a la evidencia, es decir, cubre todos los ejemplos y de manera correcta. Esto puede parecer óptimo, pero se torna ingenuo en la realidad. Por qué debe ajustarse demasiado a la evidencia, como consecuencia, el

modelo puede comportarse mal para nuevos ejemplos, ya que, el modelo en sí, es una aproximación del objetivo del aprendizaje. Por tanto, intentar aproximar demasiado hace que el modelo sea específico, poco general y, por tanto, malo con otros datos no vistos. Por otro lado, cuando la evidencia contiene ruido (errores en los atributos o incluso en las clases), el modelo podría ajustarse a los errores y perjudicar el comportamiento global del modelo aprendido (sobre ajuste u *overfitting*).

Una forma de abordar este problema es modificar los algoritmos de aprendizaje permitiendo que los modelos más generales, en este contexto, se eliminen condiciones de las ramas del árbol o de algunas reglas. A dicho procedimiento se denomina poda, los nodos que están por debajo del límite de poda se eliminan, por considerarse demasiado específicos. En Garcia. & Tsang., (2006) se presenta una técnica que realiza una poda eliminando reglas que causan clasificaciones erróneas utilizando programación genética.

Sistema de aprendizaje de reglas (Berlanga, 2010)**.** Existen varias reglas que pueden ser aplicables para al mismo ejemplo y dar la misma precisión, en general, también existen métodos que generan conjuntos de reglas que pueden ser contradictorias para algunos ejemplos, la manera de resolver esto es dar un orden a las reglas o ponderando las predicciones.

Los métodos que generan estos conjuntos van añadiendo reglas, una de detrás de otra, mientras vayan cubriendo ejemplos de una manera consistente. A diferencia de los árboles de decisión, no se sigue con las condiciones complementarias a la utilizada en la regla anterior (exclusión y exhaustividad), sino que se descartan los ejemplos ya cubiertos por las reglas ya obtenidas y con los ejemplos que quedan se empieza de nuevo. Esto hace que puedan aparecer nuevas condiciones que solapen (o no) con las anteriores. Esta manera de actuar es la que utilizan los métodos de cobertura.

El criterio de selección de las condiciones puede basarse en la pureza (la que elimine más contraejemplos) o en medidas derivadas de la información (como el Gain visto para árboles de decisión) o medidas que ponderen la precisión y el alcance (*precisión and recall*), por ejemplo la medida-f (C. J. van Rijsbergen. 1979).

A continuación, enunciamos las ventajas e inconvenientes de los algoritmos basados en partición y los algoritmos basados en cobertura.

Partición: Suelen ser más eficientes debido a que cada partición en realidad genera dos o más reglas. La poda es más sencilla.

Cobertura: Permite hacer coberturas no exhaustivas, que es interesante cuando un atributo tiene muchos valores y sólo algunos son significativos. Es menos voraz y las últimas ramas se ajustan más a los ejemplos. Esto también es una desventaja, porque las últimas ramas suelen hacer sobre ajuste (*overfitting*), es decir, intentan capturar aparentes regularidades del conjunto de entrenamiento que resultan ser ruido y el modelo es demasiado específico.

2.2. Autómatas

Intuitivamente, un autómata es un dispositivo teórico capaz de procesar una secuencia finita de símbolos de un alfabeto, cambiando su estado si procede. De entre los posibles estados que puede alcanzar un autómata se distinguen los aceptadores que indican el reconocimiento, por parte del autómata de la secuencia tratada (Castro, 2004).

2.2.1. Autómata finito determinista.

Formalmente un autómata finito determinista es una tupla:

$$D = (Q, \Sigma, \delta, q_0, F) \qquad \textbf{(6)}$$

En dónde:

Q, es un conjunto no vacío, finito de estados.

Σ, es un alfabeto finito.

$\delta: Q \times \Sigma \rightarrow Q$, es la función de transición.

$q_0 \in Q$, es el estado inicial.

$F \subseteq Q$, es el conjunto de estados de finalización o aceptadores.

La definición anterior muestra la estructura física del autómata, es decir su forma estática, en el sentido, que no determina su funcionalidad sino que muestra los elementos que lo conforman, más no su comportamiento, para agregarle dinamismo, hay que activar movimiento en la función de transición, el cual dirá cuál será el comportamiento del autómata de acuerdo al alfabeto de entrada, para esto se deberá extender la función δ al dominio $Q \times \Sigma^*$.

2.2.2. Función de transición extendida.

Dado un autómata finito determinista $D = (Q, \Sigma, \delta, q_0, F)$, se define la extensión δal dominio $Q \times \Sigma^*$, (denotada por $\hat{\delta}$) de manera inductiva en la longitud de cadenas:

Dado que $q \in Q$, definimos $\hat{\delta}(q, \varepsilon) = q$

Dados $q \in Q$, $w \in \Sigma^*$y $\sigma \in \Sigma$, definimos $\hat{\delta}(q, w\sigma) = \hat{\delta}\big(\hat{\delta}(q, w), \sigma\big)$

La definición anterior muestra la formalización del funcionamiento de un autómata: Dada una cadena $\sigma_1 \ldots \ldots \ldots \sigma_n$, la forma de actuación del autómata es leyendo cada uno de los símbolos de entrada de izquierda a derecha, y cambia su estado, si procede, en función del estado y el símbolo leído, de esta manera se forma el lenguaje del autómata, previamente cambiando el estado del símbolo a un estado de aceptación. Todos los elementos que pertenecen al conjunto de estados finitos ($q \in Q$) y deben estar interconectados por medio de alguna letra del alfabeto de entrada con otro estado de este mismo conjunto. Esto aplica que para cualquier letra del alfabeto de entrada:

$$\delta(q,\sigma) = p \mid q,p \in Q \ \wedge \ \sigma \in \Sigma \qquad (7)$$

2.2.3. Representación de autómatas finitos determinista mediante diagrama de transición.

Una representación de los Autómatas Deterministas es la dada por los diagramas de transición (Castro, 2004), En ella se representa el autómata mediante un grafo orientado sujeto a las siguientes consideraciones:

- Cada nodo del grafo tiene asociado de manera univoca un estado del autómata.
- Dos nodos, digamos p y q están unidos mediante un arco dirigido de p a q si y solo si, existe un símbolo del alfabeto digamos $\sigma \in \Sigma$ tal que $\delta(p,\sigma) = q$ dicho arco se etiqueta σ.
- Los nodos correspondientes a estados aceptadores (finalizadores) se representan mediante una doble circunferencia con el fin de distinguirlos de los restantes.
- Al nodo correspondiente al estado inicial llega una flecha sin origen concreto.

  Ejemplo 1. Considerar, por ejemplo, el autómata $A = (Q, \Sigma, \delta, q_0, F)$,

  Donde $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{0,1\}$, $F = \{q_2\}$ y la función de transiciónδ, se define mediante la tabla 2.

Tabla 3.
Función de transición δ

	0	1
$\rightarrow q_0$	q_2	q_0
q_1	q_1	q_1
$* q_2$	q_2	q_1

Nota: Función de transición δ para el autómata del ejemplo 1, observe en la tabla 2 que el estado inicial suele indicarse mediante una flecha mientras que, los estados finales, se indican con un asterisco. Tomado de (Nieto, 2011).

El diagrama de acuerdo a las especificaciones dadas puede apreciarse en la figura 2.

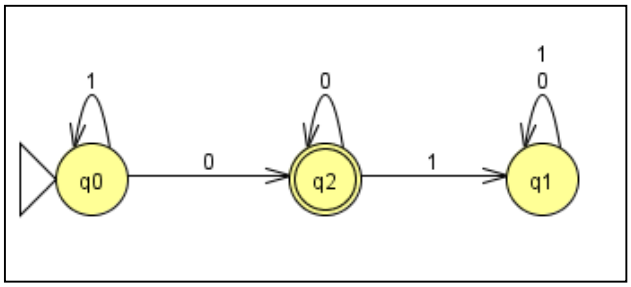


Figura 2. Representación diagrama de transición del autómata finito determinista mediante el uso de diagramas de transición del AFD
Fuente: AFD del ejemplo 1 (Nieto, 2011).

2.3. Optimización

En la actualidad diferentes áreas del conocimiento requieren de la toma de decisiones, de aquí, que los problemas deben resolverse de forma eficiente y eficaz. De ahí la importancia de la optimización, cuyo estudio es determinar la existencia de los óptimos globales de una función objetivo sobre un conjunto de soluciones factibles y las técnicas para determinarlos.

Cabe señalar que no existe un método general para la resolución de problemas, sino, que se aplican distintas técnicas que dependiendo de la naturaleza de la función objetivo y del conjunto de solución factible, definen restricciones de igualdad o desigualdad.

Uno de los problemas en la optimización global (y, básicamente, también en la naturaleza) es que a menudo no es posible determinar si la mejor solución actualmente conocida está localizada en un óptimo local o global y, por lo tanto, si la convergencia es aceptable. En otras palabras, generalmente no está claro si el proceso de optimización se puede detener, si debería concentrarse en refinar el óptimo actual, o si debería examinar otras partes del espacio de búsqueda. Esto, por supuesto, solo puede volverse engorroso si hay múltiples óptimos (locales).

En este capítulo se profundizará en los conceptos de optimización en general, así como la optimización multi objetivo en particular, las técnicas existentes para dar solución a problemas de optimización multi objetivo, que es objeto de estudio de esta tesis.

2.3.1. Concepto de optimización.

La optimización se puede explicar cómo el proceso de encontrar la mejor solución posible a un problema determinado (Lobo & Castro, 2011), (Martí, 2003) Introduce un concepto más preciso, de optimización, como el proceso cuyo finalidad es encontrar el valor de unas variables de decisión para que una determinada función objetivo alcanza su valor máximo o mínimo, sujetas a unas restricciones.

En síntesis, la optimización es el estudio matemático de los problemas, En general, la formulación es la siguiente:

$$Optimizar\ f\ (x) \qquad \textbf{(8)}$$
$$Sujeto\ a \quad x \in M$$

Se trata de encontrar los valores de x en un conjunto M, llamado conjunto factible, que maximiza o minimiza la función *f*, llamada función objetivo Y, además, el conjunto factible está definido por medio de condiciones de igualdad (g(x=0)) o desigualdad (h (x≤0)) que se denominan restricciones.

Para una mejor interpretación de la teoría de optimización en la solución de problemas, se requiere definir los siguientes términos implicados en el proceso.

Variables de decisión:

Son los valores que se modifican para resolver el problema. Su interpretación es la siguiente (Cagnina, 2010):

Son el conjunto de *n* parámetros (de entada para la función objetivo) cuyos valores arrojan una posible solución al problema; Estas variables pueden ser entera, real, o cualquier combinación entre ellas. Cada uno de estos parámetros es denotado como $x_i = 1, 2\ldots\ldots, n$. El vector $\underset{X}{\rightarrow}$ de *n* variable se representará de la siguiente forma:

$$\vec{X} = [X_1, X_2, \ldots\ldots X_n] \qquad \textbf{(9)}$$

Función objetivo: Forma el criterio de evaluación de las variables de decisión, que nos dirá la calidad de la solución. La definición de la función objetivo está dada por la naturaleza del problema. La función objetivo se define como $f\left(\underset{X}{\rightarrow}\right)$. Los problemas pueden tener una sola función o varias. Cuando se tiene una sola función, se considera que es un problema de optimización mono-objetivo y cuando se evalúan más de dos funciones objetivo se habla de un problema de optimización multi-objetivo (Cagnina,2010).

Restricciones o limitaciones: Las restricciones se expresan en ecuaciones de igualdad o desigualdad,generalmente involucra dos componentes, una función y una relación constante, finalmente para que una solución sea considerada como factible se deben cumplir todas las restricciones formuladas, en el caso que no se formulen ninguna restricción todas las soluciones de las función objetivo se evaluaran para resolver el problema (Cagnina, 2010).

De hecho, existe la posibilidad de hallar varias soluciones consideradas como factibles para optimizar un mismo problema, es decir, que se hallen varios óptimos que cumplen con todas las restricciones del proceso. Sin embargo, la solución que mejor se adapte a las circunstancias del problema, es la que finalmente es la considerada como la solución óptima global, de lo anterior se puede concluir que las soluciones que no cumplan con las restricciones son consideradas soluciones no factibles.

A continuación, se muestra la estructura general de un modelo matemático (también conocido como modelo matemático de programación) y se puede representar de la siguiente manera:

Encontrar x tal que:

$$Optimizar\ [maximizar/minimizar]\ f(x) \qquad \textbf{(10)}$$

Sujeto a:

$$g_i(x) \le gb_i, \quad i = 1,\ldots,m \qquad \textbf{(11)}$$
$$h_j(x) \le hb_j, \quad j = 1,\ldots,p$$

$$x \ge 0 \qquad \textbf{(12)}$$

Donde la función objetivo f es una función de una única variable x, y la restricción de las funciones g_i y h_i son las funciones generales de la variable (de lo contrario se expresa como una variable de decisión desconocida, o algunas veces como un parámetro) $\boldsymbol{x} \in \mathbb{R}^{\boldsymbol{n}}$. Los lados de la derecha, gb_i y hb_j, suelen ser las constantes conocidas por problemas deterministas. La restricción de no negatividad, $x \ge 0$ es necesario para muchos problemas de orden práctico (ya que muchas variables no puede ser negativo) y para la solución de muchos enfoques (suposición por defecto). El estándar por encima de puede variar el modelo de la siguiente manera:

- Contiene los límites superior e inferior de x en lugar de una restricción de no negatividad,
- Contiene un límite superior e inferior de x en lugar de cualquier otra restricción, y
- Por encima del modelo estándar, con o sin **(10)** y **(11)**, con múltiples variables.

Supongamos $\bar{x}$ representa un conjunto de variables, donde $\bar{x} = (x_1, x_2, \ldots, x_n)$ entonces el modelo anterior puede ser reescrito por múltiples variables de la siguiente manera:

Encontrar $_x$ tal que:

$$Optimizar\ [maximizar/minimizar]\ f(\bar{x}) \quad \textbf{(13)}$$

Sujeto a:

$$g_i(\bar{x}) \leq gb_i, \quad i = 1, \ldots, m \quad \textbf{(14)}$$
$$h_j(\bar{x}) \leq hb_j, \quad j = 1, \ldots, p$$

$$\bar{x} \geq 0 \quad \textbf{(15)}$$

2.3.2. Optimización mono-objetivo.

Consiste en aquellos problemas, cuya función tiene un solo objetivo. Esto quiere decir que el proceso consiste en optimizar una sola función objetivo teniendo en cuenta una restricciones, que están basadas en restricciones de del mundo real(Donoso, Yezid; Fabregat, 2007), (Sui. X & Leung. H, 2008). Un problema de optimización mono objetivo es expresado de la siguiente forma:

$$Optimizar\ [maximizar/minimizar]\ f(X) \quad \textbf{(16)}$$

Sujeto a:

$$H(X) = 0 \quad \textbf{(17)}$$

$$G(X) \leq 0 \qquad \textbf{(18)}$$

En este caso, la función a optimizar (maximizar o minimizar) es $f(X)$, donde el vector X es el conjunto de variables independientes. Las funciones $H(X)$ y $G(X)$ son las restricciones del modelo, para este problema existen tres conjuntos soluciones:

El **conjunto universal**, el cual está compuesto por todos los posibles valores de X sean factibles o no.

El **conjunto de soluciones factibles**, el cual está compuesto por todos los valores de X que cumplen con las restricciones $H(X)$ y $G(X)$. Por último,

El **conjunto de soluciones optimas**, el cual está compuesto por todos los valores de X que, en adición de ser factibles, alcanza el valor óptimo (mínimo o máximo) de la función $f(X)$, sea en un intervalo específico $[a, b]$ ó en un contexto global $(-\infty, \infty)$. Los tres conjuntos mencionados se pueden apreciar en la figura 3.

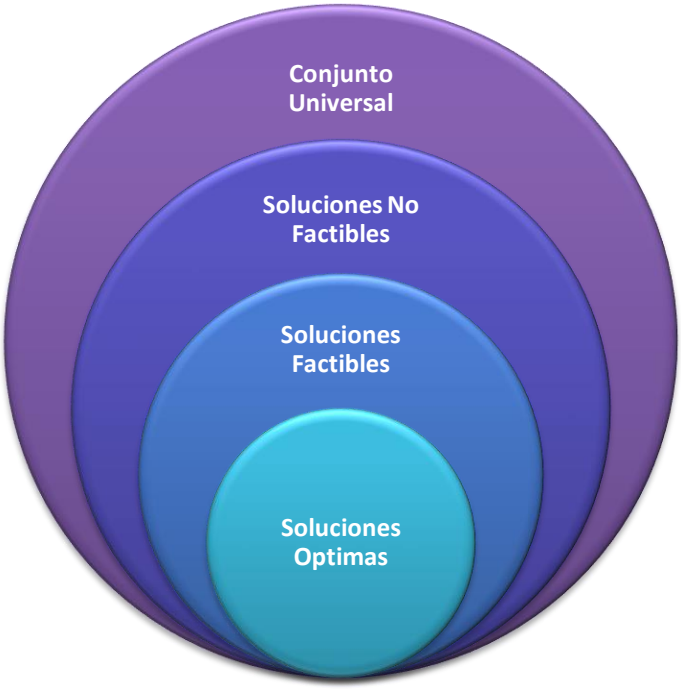


Figura 3. Conjuntos soluciones para un problema mono-objetivo
Nota: Los tres posibles conjuntos soluciones en un problema mono-objetivo, lo ideal es alcanzar el conjunto de soluciones óptimas (Ruiz, 2013).

2.3.3. Optimización multi-objetivo.

Consiste en aquellos problemas, cuya función tiene más de un objetivo. Esto quiere decir que el proceso consiste en optimizar más de una función objetivo teniendo en cuenta una restricciones (Donoso, Yezid; Fabregat, 2007), que están basadas en restricciones de del mundo real. Debido a que existen varias funciones a optimizar, en este caso, se dice que el problema de optimización es multi-objetivo. Un problema de optimización multi-objetivo es expresado de la siguiente manera:

$$Optimizar\ [maximizar/minimizar]\ F(X) = \{f_1(X), f_2(X), \dots, f_n(X)\} \quad \textbf{(19)}$$

Sujeto a: **(20)**

$$H(X) = 0$$

$$G(X) \leq 0 \quad \textbf{(21)}$$

En este caso, las funciones a optimizar (cada una puede ser maximizada o minimizada) son el conjunto de funciones $F(X)$, donde X es el conjunto de variables independientes. Las funciones $H(X)$ y $G(X)$ son las restricciones del modelo. Debido a que existe más de una función a optimizar, es posible que la mejora de una perjudique al resto de funciones.

2.3.3.1. Frente de Pareto (fp).

La interpretación de un problema de optimización multi-objetivo es indiscutiblemente diferente a la de un problema mono-objetivo. En el enfoque mono-objetivo se busca la mejor solución de todas (Gen, Mitsuo; Cheng, 2000). Observe el conjunto de soluciones de la figura 4, en el caso de que las soluciones correspondieran a un problema de minimización, la mejor solución sería f_3 por ser el menor del conjunto, pero, si el problema fuese de maximización, la mejor solución sería f_2 por ser el mayor del conjunto. En cambio, en un problema multi-objetivo No hay un punto de referencia exacto por el cual se pueda decir que una solución es mejor que otra. Observe el conjunto de soluciones de la figura 5, para un problema con dos objetivos; en el caso de que las soluciones correspondieran a un problema de minimización, no se puede determinar entre las dos primeras soluciones cual es mejor ya que $f_{11} > f_{21}$ y $f_{12} < f_{22}$, sin embargo, las dos primeras soluciones son mejores que la tercera, ya que para este caso en particular ambas funciones son menores. En este caso se dice que la función 3 es dominada por la solución 1 y la solución 2.

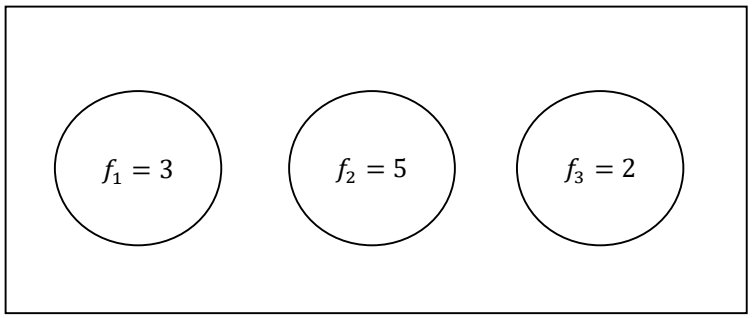


Figura 4. Conjunto de soluciones mono-objetivo
Nota: Tres soluciones para un problema particular mono-objetivo. Si se tratase de un problema de maximización la mejor solución sería f2. En el caso contrario, si el problema fuese de minimización, la mejor solución sería f_2. *En el caso contrario, si el problema fuese de minimización, la mejor solución sería* f_3*(Niño, 2010).*

.

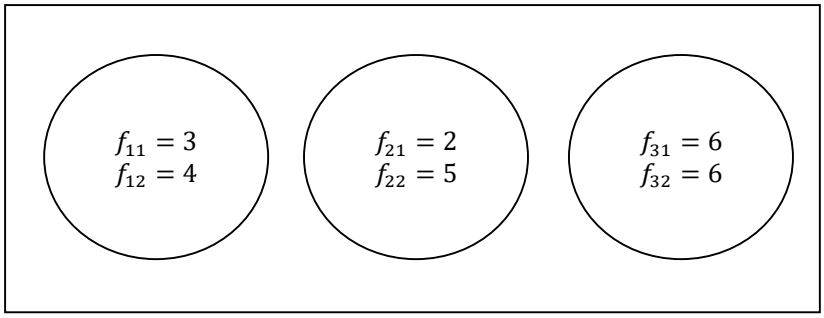


Figura 5. Conjunto de soluciones multi-objetivo
Nota: Tres soluciones para un problema particular con dos objetivos (Niño, 2010).

Formalmente, se dice que una solución a**domina a** una solución b si y solo si:

$$(\forall f_{ai} \in a)(\forall f_{bi} \in b)(f_{ai}\ es\ mejor\ que\ f_{bi}) \quad \textbf{(22)}$$

Formalmente, se dice que una solución a**no domina a** una solución b si y solo si:

$$(\forall f_{ai} \in a)(\exists f_{bi} \in b)(f_{ai}\ no\ es\ mejor\ que\ f_{bi}) \quad \textbf{(23)}$$

El conjunto solución de un problema multi-objetivo se conoce con el nombre de **soluciones no dominadas** (Donoso, Yezid; Fabregat, 2007)**,** en el caso particular de dos y

tres objetivos, las soluciones no dominadas forman una gráfica conocida con el nombre de **Frente de Pareto (FP)** (Ruiz, 2013)**,** un ejemplo de FP puede ser apreciado en la figura 6.

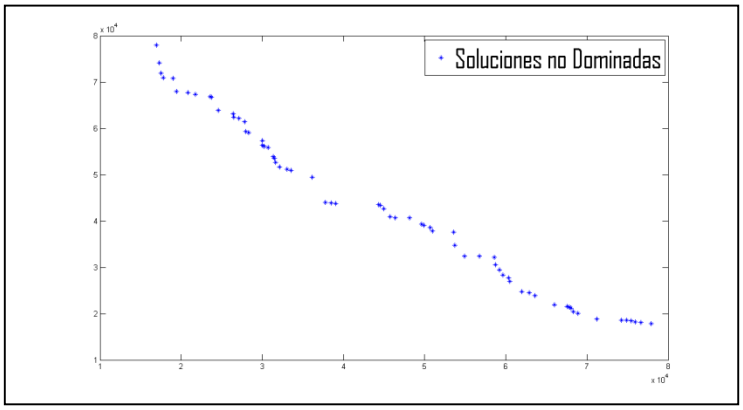


Figura 6. FP Bi-Objetivo
Nota: Conjunto de soluciones no dominadas para un problema en particular con dos objetivos (Nieto, 2011).

2.3.3.2. Métricas.

Las métricas son utilizadas para medir la calidad de soluciones no dominadas. La calidad de las soluciones dependerá del tipo de métrica utilizado. A continuación, se muestra algunas métricas.

2.3.3.2.1. Generación de vectores no dominados (GVND)

La métrica GVND(Tovar, Coronell, & Donoso, 2007),índica el número de elementos no dominados generados por el algoritmo, es decir nos muestra cuántos elementos hay en el FP, esto es:

$$GVND = |FP_{real}| \quad \textbf{(24)}$$

2.3.3.2.2. Distancia generacional (GD).

La métrica GD (Tovar et al., 2007), nos índica qué tan lejos está el FP Aproximado ($\boldsymbol{FP_{aprox}}$) del FP Real ($\boldsymbol{FP_{real}}$), esto es:

$$DG = \frac{\sqrt{\sum_{i=1}^{|FP_{aprox}|} d_i}}{|FP_{aprox}|} \quad \textbf{(25)}$$

Donde d_i es la mínima distancia Euclidiana entre el i-ésimo elemento del FP aproximado con los elementos del FP real.

2.3.3.2.3. Distancia generacional inversa (IGD)

La métrica GD nos índica qué tan lejos está el FP$\boldsymbol{A}$ de un FP de referencia $\boldsymbol{A}^*$, esto es: (Zitzler; Thiele; Laumanns; Fonseca; Fonseca, 2003),

$$IGD(A, A^*) = \frac{1}{|A^*|}\sum_{i=1}^{|A|} d_i \quad \textbf{(26)}$$

Donde d_i es la mínima distancia Euclidiana entre el i-ésimo elemento del FP$\boldsymbol{A}$ con los elementos del FP de Referencia $\boldsymbol{A}^*$.

2.3.3.2.4. Espaciamiento.

Espaciamiento (Tovar et al., 2007), la métrica S, nos índica qué tan bien están distribuidas las soluciones sobre el FP. Matemáticamente se define como:

$$S = \sqrt[2]{\left(\frac{\sum_{i=1}^{|FP_{conocido}|}(\bar{d} - d_i)}{|FP_{conocido}|}\right)} \quad \textbf{(27)}$$

Donde $d_i = min_j\left(\sum_{k=1}^{m}\left|f_k^i(x_i) - f_k^j(x_j)\right|\right)$con $i \neq j$ y $1 \leq i, j \leq |FP_{conocido}|$ y $\bar{d}$es el promedio de todos los d_i.

Para esta métrica un valor de 0 significa que todos los elementos de $FP_{conocido}$ están equidistantemente espaciados. Por lo tanto, entre menor sea el valor de S, mejor es la distribución de los elementos del $FP_{conocido}$.

2.3.3.3. Técnicas clásicas.

Las técnicas clásicas de optimización multi-objetivo funcionan aproximando las soluciones del problema planteado mediante la optimización de un problema mono-objetivo(Tovar et al., 2007). A continuación, se mencionan dos de las más conocidas:

2.3.3.3.1. *Suma con pesos.*

Este método consiste en crear un modelo mono-objetivo mediante la ponderación de pesos en n funciones objetivos del problema. A través de la suma con pesos, la expresión 4 puede ser redefinida como:

$$Optimizar\ [maximizar/minimizar]F'(X) = \sum_{i=1}^{n} \alpha_i \times f_i \qquad \textbf{(28)}$$

Sujeto a:

$$H(X) = 0 \qquad \textbf{(29)}$$

$$G(X) \leq 0 \qquad \textbf{(30)}$$

$$\sum_{i=1}^{n} \alpha_i = 1, \qquad 0 \leq \alpha_i \leq 1 \qquad \textbf{(31)}$$

Como puede apreciarse en **(28)**, $F'(X)$ es una combinación lineal de $F(X)$ debido a la restricción **(27)**.

***2.3.3.3.2. ε** – restricción.*

Esta técnica consiste en crear un modelo mono-objetivo optimizando solo una función de 4 y convirtiendo el resto de funciones en restricciones del problema. A través de ε – Restricción la expresión (19) se convierte en:

$$Optimizar\ [maximizar/minimizar] f_i(X) \quad \textbf{(32)}$$

Sujeto a:

$$H(X) = 0 \quad \textbf{(33)}$$

$$G(X) \leq 0 \quad \textbf{(34)}$$

$$f_k(X) \leq \varepsilon_k \ , k = 1, \dots, n \ y \ i \neq k \quad \textbf{(35)}$$

En este caso la función $f_i(X)$ es la única a ser optimizada. El resto de funciones objetivos se convierten en restricciones del problema. Para cada $f_k(X)$ existe un valor $\varepsilon_k$que la acota. Cuando es alterado algún ε_k se obtienen nuevas soluciones para (28).

2.3.4. Teoría combinatoria.

La combinatoria es la parte de las matemáticas que estudia las diversas formas de realizar agrupaciones con los elementos de un conjunto.

Existen distintas formas de realizar las agrupaciones y se clasifican según se repitan o no sus elementos, en el caso de las combinaciones, según el orden de colocación en que se hallan los elementos, en el caso de las permutaciones.

Combinaciones: son aquellas formas de agrupar los elementos de un conjunto teniendo en cuenta lo siguiente:

- No influye el orden en que se colocan

- Se pueden repetir sus elementos.

Existen dos tipos: combinaciones sin repetición y combinaciones con repetición.

Combinaciones sin repetición: Combinaciones sin repetición de n elementos tomados de p en p se define como las agrupaciones formadas con p elementos distintos, eligiéndose entre los n elementos, esto significa que son las todas las diferentes agrupaciones que pueden formarse de tal manera, que dichas agrupaciones difieren entre sí en al menos un elemento, se denota de la siguiente manera: $C_n^P = \left\langle {n \atop p} \right\rangle = \frac{n!}{(n-p)!p!}$

Combinaciones con repeticiones: Combinaciones con repeticiones de n elementos tomados de p en p se define como las distintas agrupaciones formadas con p elementos que pueden repetirse eligiéndose entre los n elementos, esto significa que hay una variación distinta a otra si difieren en algún elemento. Por otro lado, no influye el orden de colocación de sus elementos.

El número de combinaciones que se pueden obtener se calcula mediante la fórmula: $CR_n^p = \left\langle {n+p-1 \atop p} \right\rangle$

Permutaciones. Son aquellas formas de agrupar los elementos teniendo en cuenta lo siguiente:

- Influye el orden en que se colocan.
- Cuando todos los elementos que se disponen son distintos son permutaciones sin repetición.
- Cuando hay disposición de elementos repetidos son permutaciones con repetición.

Existen dos tipos: permutaciones sin repetición y permutaciones con repetición.

Permutaciones sin repetición: las permutaciones sin repetición de n elementos se define como las distintas formas de ordenar todos los elementos distintos, se difiere solo en el orden de colocación de sus elementos, se denota de la siguiente manera: $P_n = n!$

Permutaciones con repetición: las permutaciones con repetición de n elementos se define cuando existen elementos repetidos. Se denota de la siguiente manera: $PR_n^{a,b,c} = \frac{n!}{a!b!c!}$

2.3.5. Optimización combinatoria.

El desafío que enfrentan los expertos en datos es sin duda los problemas de decisión. La optimización combinatoria es un campo de las matemáticas aplicadas, que combina técnicas de combinatoria, programación lineal y la teoría de algoritmos para resolver este tipo de problemas (Hincapie et al., 2004), uno de los problemas de mayor interés es el problema del Agente Viajero (Traveling-Salesman-Problem). Consiste en que partiendo de una ciudad de un conjunto de ciudades, se vuelva al origen pasando exactamente una vez por cada una de las ciudades con el menor costo posible, sin duda es uno de los problemas más estudiados a nivel mundial por su alto campo de acción: administración de depósitos (Oberlin.P, Rathinam.S, & Darbha.S, 2009).

El modelo matemático que representa el problema es el siguiente:

$$optimizar[minimizar|maximizar]\, Z(x) = \sum_{(i,j)\in A} c_{ij}\, x_{ij} \qquad \textbf{(36)}$$

Sujeto a:

$$\sum_{\{i:(i,j)\in A\}} x_{ij} = 1 \;\forall\, j \;\in V \qquad \textbf{(37)}$$

$$\sum_{\{j:(i,j)\in A\}} x_{ij} = 1 \; \forall \; i \; \in V \qquad \textbf{(38)}$$

$$\sum_{\{(i,j)\in A: i\in U, j\in (V-U)\}} x_{ij} \geq 1 \; 2 \leq |U| \leq |V| - 2 \qquad \textbf{(39)}$$

2.3.6. Optimización multi-objetivo y autómatas.

Un Autómata Finito Determinista de Intercambio (AFDI) propuesto por Niño. & Ardila., (2009), es un grafo que permite representar el espacio de soluciones factibles de un problema de naturaleza combinatoria en los cuales el orden de los elementos importa y no se permiten repeticiones. Formalmente, un AFDI se define como:

$$M = (\overline{X_0}, f(\bar{X}), Q, \Sigma, \delta, q_0, F) \qquad \textbf{(40)}$$

Dónde:

$\overline{X_0}$ Es el vector que contiene el orden original de los elementos asociados al problema representado.

f($\bar{X}$) Es la función que se desea optimizar en el problema modelado.

Q, es el conjunto de estados que conforman el autómata, cada estado $q_k \in Q$ representa una solución al problema modelado. La estructura de cada estado viene conformada de la siguiente manera:

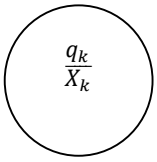


Figura 7. Estructura de un estado de un AFD I
Nota: Cada estado que compone un AFDI está compuesto por un orden específico de los elementos modelados (Nieto, 2011).

.

$\overline{X_k}$, corresponde a un orden específico de los elementos asociados al problema modelado y así mismo se convierte en una solución factible al problema propuesto.

Σ es el alfabeto finito de entrada. Este contiene todas las posibles combinaciones entre los índices del vector de entrada agrupándolos en parejas, debido a que estos son distintos, las parejas de Σ vendrían dadas por tuplas que cumplen la condición:

$$\{(k,j)|k = 1,2 \dots n, j = k+1, k+2, .., n\} \quad \textbf{(41)}$$

Donde n es el número de elementos del vector de entrada.

δ, es la función de transición la cual toma un estado q, $q \in Q$, y una tupla de Σ y devuelve un nuevo estado t, $t \in Q$, esto es $\delta\big(q,(i,k)\big) = t$, en donde $(i,k) \in \Sigma$. Es importante resaltar que, debido a que los AFD I provienen de los AFD , por 3.2.1.1, cada estado $q \in Q$ debe tener transición con todos los símbolos de Σ. q_0 es el estado inicial del AFD I. Este contiene una solución inicial para el problema modelado.F, es el conjunto de estados de finalización, para los AFD I todos los elementos de Q son elementos de F.

Ejemplo 2. Supóngase el vector de entrada $\overline{\mathrm{X}} = (1,2,3)$ y la función objetivo (Niño. & Ardila., 2009):

$$f(\bar{X}) = 0{,}3 * x_1 + 0{,}2 * x_2 + 0{,}1 * X_3 \quad \textbf{(42)}$$

Para la construcción del AFDI equivalente a estos datos, procedemos a hacer lo siguiente:

Establecer el estado inicial: En este caso nuestro estado inicial q_0 contendrá el vector de entrada $\bar{X} = (1,2,3)$, por lo tanto $\bar{X} = \overline{X_0}$.

Establecer el conjunto Σ: Por la ecuación 41 sabemos que $\Sigma = \{(1,2), (1,3), (2,3)\}$.

Establecer la función δ: Debido a que los AFDI exigen que los estados tengan transiciones con todos los símbolos del alfabeto, comenzamos por el estado inicial el cual es q_0, cada estado nuevo lo nombramos como q_k, luego hacemos transiciones con cada q_k hasta agotarlos todos como podemos apreciarlo en la tabla 4.

Tabla 4.

Función δ para el AFD I con vector de entrada X¯=(1,2,3)

Función δ		
$\delta(q_0,(1,2)) = (2,1,3) = \boldsymbol{X_3 = q_3}$	$\delta(q_0,(1,3)) = (3,2,1) = \boldsymbol{X_5 = q_5}$	$\delta(q_0,(2,3)) = (1,3,2) = \boldsymbol{X_1 = q_1}$
$\delta(q_1,(1,2)) = (2,3,1) = \boldsymbol{X_4 = q_4}$	$\delta(q_1,(1,3)) = (3,1,2) = \boldsymbol{X_2 = q_2}$	$\delta(q_1,(2,3)) = (1,2,3) = \boldsymbol{X_0 = q_0}$
$\delta(q_2,(1,2)) = (3,2,1) = \boldsymbol{X_5 = q_5}$	$\delta(q_2,(1,3)) = (1,3,2) = \boldsymbol{X_1 = q_1}$	$\delta(q_2,(2,3)) = (2,1,3) = \boldsymbol{X_3 = q_3}$
$\delta(q_3,(1,2)) = (1,2,3) = \boldsymbol{X_0 = q_0}$	$\delta(q_3,(1,3)) = (2,3,1) = \boldsymbol{X_4 = q_4}$	$\delta(q_3,(2,3)) = (3,1,2) = \boldsymbol{X_2 = q_2}$
$\delta(q_4,(1,2)) = (1,3,2) = \boldsymbol{X_1 = q_1}$	$\delta(q_4,(1,3)) = (2,1,3) = \boldsymbol{X_3 = q_3}$	$\delta(q_4,(2,3)) = (3,2,1) = \boldsymbol{X_5 = q_5}$
$\delta(q_5,(1,2)) = (3,1,2) = \boldsymbol{X_3 = q_3}$	$\delta(q_5,(1,3)) = (1,2,3) = \boldsymbol{X_0 = q_0}$	$\delta(q_5,(2,3)) = (2,3,1) = \boldsymbol{X_4 = q_4}$

Nota: Como puede observarse en la tabla 3 todos los estados tienen transiciones con todos los símbolos del alfabeto y en consecuencia el AFD – I contiene todas las posibles permutaciones para los elementos del vector de entrada $\bar{X}$. Tomado de Ruiz (2013)

Establecer los valores obtenidos para cada evaluación de los $\overline{X_k}$ en la función objetivo. El siguiente paso es calcular el valor de la evaluación en la función objetivo de cada $\overline{X_k}$, con lo cual obtenemos la tabla 3.

Tabla 5.
Evaluación de los vectores en la función objetivo

Estado $\boldsymbol{q_k}$	Vector $\overline{X_k}$	Valor $\boldsymbol{f(\overline{X_k})}$
$\boldsymbol{q_0}$	(1,2,3)	1
$\boldsymbol{q_1}$	(1,3,2)	1.1
$\boldsymbol{q_2}$	(3,2,1)	1.4
$\boldsymbol{q_3}$	(2,1,3)	1.1
$\boldsymbol{q_4}$	(2,3,1)	1.3
$\boldsymbol{q_5}$	(3,2,1)	1.4

Nota: El AFDI de la figura 8, representa el autómata de este ejemplo. Nótese que cada estado tiene transición con todos los elementos de Σ lo cual representa el intercambio llevado a cabo en el vector $\overline{X_k}$ asociado al estado $q_k$para alcanzar el nuevo estado q_t cuyo vector es $\overline{X_t}$. Tomado de (Nieto, 2011).

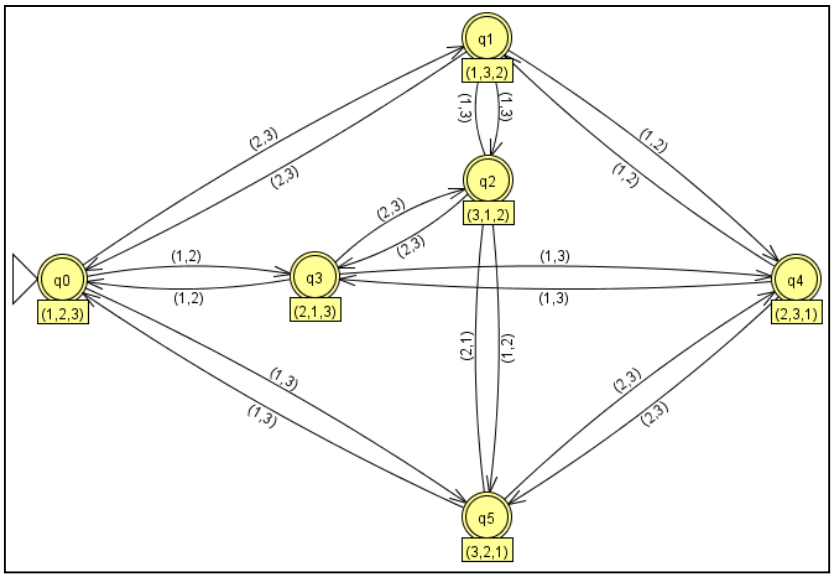


Figura 8. Representación del AFDI para el vector de entrada $\overline{X}$ = (1,2,3)
Nota: Cada estado representa una solución factible (Nieto, 2011).

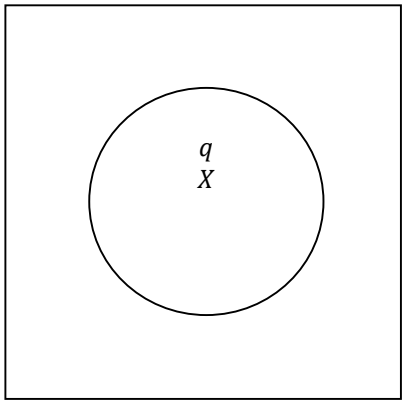


Figura 9. Estructura de un estado q∈Q
*Nota: Cada estado posee una solución*X *distinta(Nieto, 2011).*

2.3.7. Autómata finito determinista multi-objetivo.

Un Autómata Finito Determinista Multi-objetivo (AFDM) es una estructura de datos que permite el modelado del espacio de soluciones factibles de un problema combinatorio bajo un esquema multi-objetivo.

Formalmente, un AFDM se define como:

$$M = (Q, \Sigma, \delta, Q_0, F(X)) \quad \textbf{(43)}$$

2.3.7.1. Q Conjunto de Estados**.**

Q es el conjunto de Estados del AFDM. Cada estado Q contiene un vector solución X factible del problema combinatorio modelado.XContiene un orden de las variables decisión del problema combinatorio.

La estructura de un estado $q \in Q$ se aprecia en la figura 9.

No existen dos estados con el mismo X.

Definición 1.X_q. Se define X_q como el vector de la solución X del estado $q \in Q$.

Ejemplo 1: Sea una empresa que tiene tres máquinas que realizan un trabajo cualquiera, se tienen tres procesos P_1, P_2 y P_3 cuya duración es de 10, 50 y 5 minutos respectivamente, si los procesos

se ejecutan en paralelo y pueden utilizar cualquiera de las máquinas disponibles, ¿De cuantas maneras se pueden asignar las máquinas a procesos distintos para que se ejecuten?

Fácilmente esta situación puede presentarse a diario en cualquier empresa de producción que trabaje con máquinas y procesos en paralelo, supongamos que el vector $\overline{\boldsymbol{P}} = (\boldsymbol{P_k}, \boldsymbol{P_i}, \boldsymbol{P_j})$representa la solución máquina 1 ejecuta $\boldsymbol{P_k}$, máquina 2 ejecuta $\boldsymbol{P_i}$, y máquina 3 ejecuta $\boldsymbol{P_j}$, por lo tanto el conjunto de estados Qque representa el conjunto de soluciones factibles asociados a este problema son los representados en la figura 10.

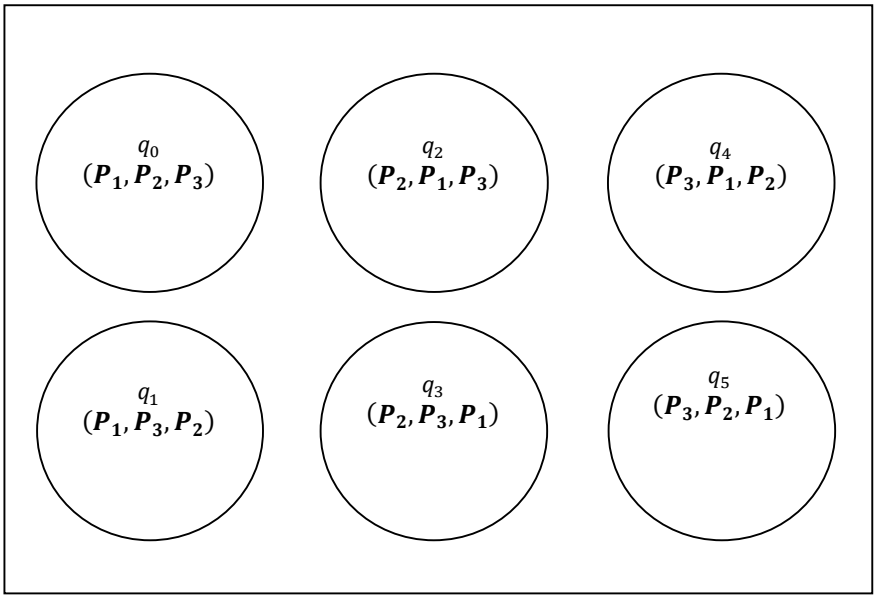


Figura 10. Representación del espacio de soluciones factibles
Nota: El problema del ejemplo 1. Cada estado posee una manera diferente de asignar las máquinas a los procesos (Nieto, 2011).

Como puede apreciarse en la figura 17, por la definición 1 tenemos que $X_{q0} = (P_1, P_2, P_3)$,$X_{q1} = (P_1, P_3, P_2)$,$X_{q2} = (P_2, P_1, P_3)$,$X_{q3} = (P_2, P_3, P_1)$,$X_{q4} = (P_3, P_1, P_2)$ y $X_{q5} = (P_3, P_2, P_1)$.

Debido a que en cada estado X_q es diferente, y como en un AFDM el número de estados es igual al número de todas las posibles soluciones, si no existen restricciones en el problema, existen $n!$ estados, esto es:

$$\#_{estados}(Q) = n! \quad \textbf{(44)}$$

2.3.7.2. Σ Alfabeto Finito de Entrada.

El conjuntoΣse conoce como el alfabeto finito de entrada.ΣEstá compuesto por los elementos que cumplen la condición:

$$\{(k,j)|k = 1,2 \dots n, j = k+1, k+2, .., n\} \quad \textbf{(45)}$$

Esto implica que Σ contiene todas las posibles maneras de intercambiar los elementos de un vector. Consecuentemente, Σ indica todas las posibles maneras de intercambiar los elementos de cualquier X_q.

Teorema 1. Dado un AFDM cuyos vectores solución son de norma n, el número de elementos del conjunto Σ es igual a:

$$\#_{elementos}(\Sigma) = \frac{n \times (n-1)}{2} \quad \textbf{(46)}$$

Demostración 1.

Por la definición de subgrupos, si agrupamos los elementos en grupos de a dos (k = 2), tendríamos la siguiente expresión:

$$nC_2 = \frac{n!}{(n-2)! \times 2!} \quad \textbf{(47)}$$

Aplicando la definición de factorial de manera recurrente en la ecuación **(37)** se obtiene:

$$nC_2 = \frac{(n-2)! \times (n-1) \times n}{(n-2)! \times 2!} \quad \textbf{(48)}$$

Cancelando los $(n-2)!$ y reemplazando el valor de 2! = 2 en la ecuación **(38)** se obtiene:

$$nC_2 = \frac{(n-1) \times n}{2} \quad \textbf{(49)}$$

Y con ello queda demostrado el teorema 1.

Ejemplo 2. El conjunto Σ para el problema del ejemplo 1 es:

$$\Sigma = \{(P_1, P_2), (P_1, P_3), (P_2, P_3)\} \quad \textbf{(50)}$$

La cantidad de elementos concuerda con la ecuación la propuesta por el teorema 1:

$$\#_{elementos}(\Sigma) = \frac{3 \times (3-1)}{2} = 3 \quad \textbf{(51)}$$

2.3.7.3. δ Función de transición.

La función de transición δdescribe el comportamiento del AFDM.δPermite el desplazamiento entre los diversos estados del AFDM utilizando elementos deΣ, esto es:

$$\boldsymbol{\delta(q,a) = r/q, r \in Q \wedge q \neq r \wedge a \in \Sigma} \quad \textbf{(52)}$$

Como puede apreciarse, la función descrita en 42, obtiene un elemento de Σ para pertubar los elementos del vector X_q, es decir, intercambiar los elementos del vector y consecuentemente, caer en otro estado con solución X_r. La función es definida formalmente como:

$$\boldsymbol{\delta: Q \rightarrow Q} \quad \textbf{(53)}$$

Ejemplo 3. Retomando el ejemplo 1, la función δ definida para el estado q_0 es: Sobre cada nuevo estado destino obtenido al aplicar la función δ (en este caso q_1,q_2 y q_5), debe aplicarse la misma función con el objetivo de alcanzar nuevos estados. El proceso termina cuando no se obtienen nuevos estados. Una vez se han obtenido todos los estados, se puede afirmar que se ha recorrido el espacio de soluciones factibles del problema modelado por medio del AFDM, lo que equivaldría a una búsqueda exhaustiva.

Tabla 6.
Función δ para q_0 del AFDM del ejemplo 1.

Estado origen q	**Elemento de Σ**	**$\delta(q,a)/\ q \in Q a \in \Sigma$**	**Estado destino r**
q_0	(P_1, P_2)	$\delta(q_0, (P_1, P_2))$	q_2
q_0	(P_1, P_3)	$\delta(q_0, (P_1, P_3))$	q_5
q_0	(P_2, P_3)	$\delta(q_0, (P_2, P_3))$	q_1

Nota: Debido a que los AFDM provienen de los AFD, todos los estados deben tener transición con todos los elementos del alfabeto (Nieto, 2011).

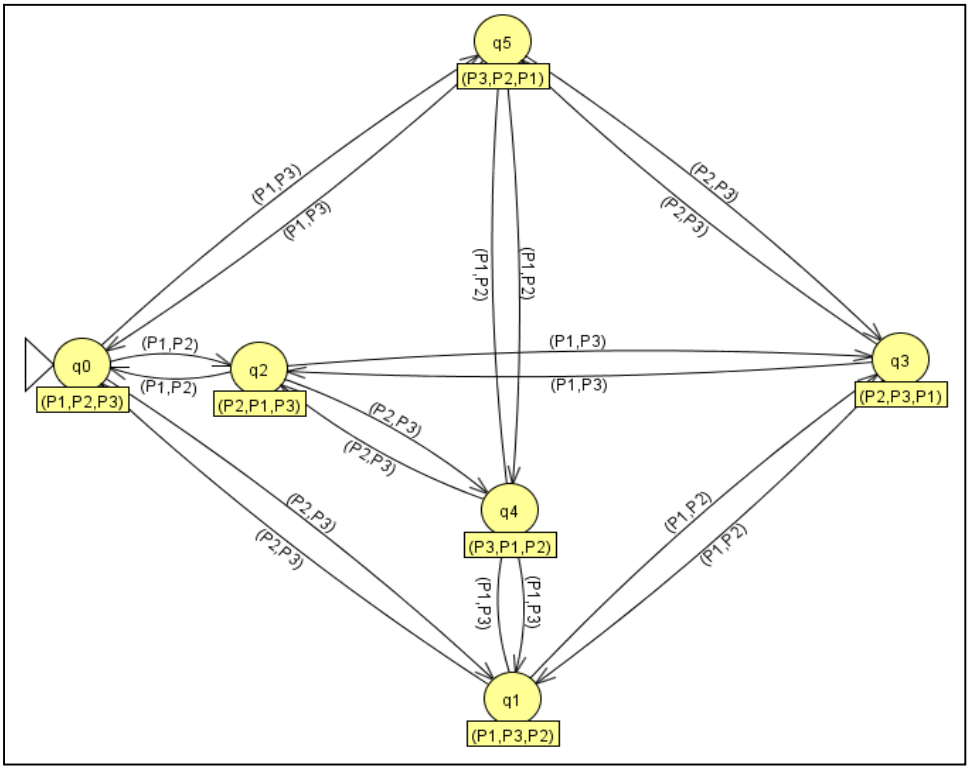


Figura 11. AFDM construido a partir del estado inicial q0
Nota: Del ejemplo 1.El AFDM es construido al aplicar la definición de δ sobre cada nuevo estado encontrado (Nieto, 2011).

2.3.7.4. Q_0 conjunto de estados iniciales.

El conjunto de estados inícialesQ_0 contiene los estados por el cual se comenzará a generar el autómata. En otras palabras, Q_0 representa las soluciones iníciales del problema. Así pues en el ejemplo 3 Q_0 solo contenía un elemento igual al estado Q_0 .

2.3.7.5. $F(X)$ conjunto de funciones objetivos.

La función $F(X)$ es el conjunto de funciones objetivos del problema, si el problema tiene n objetivos, $F(X)$ es:

$$\boldsymbol{F(X) = \{f_1(X), f_2(X), \dots, f_n(X)\}} \quad \textbf{(54)}$$

Esta función se encuentra definida formalmente como:

$$\boldsymbol{F: R^n \rightarrow R^m | \; n \geq 1 \wedge m \geq 2} \quad \textbf{(55)}$$

$F(X)$ Recibe como argumento de entrada una solución X_q. En **(45)**, n representa el número de variables decisión del problema combinatorio y $m$representa el número de funciones objetivo. Por lo tanto $F(X)$ toma las variables decisión y las evalúa en cada función objetivo devolviendo una m – tupla.

Ejemplo 4. Supóngase el siguiente conjunto de funciones bi-objetivo (2 objetivos):

$$F(X) = \left\{ f_1(X) = \sum_{i=1}^{3} i * X_i \, , f_2(X) = \sum_{i=1}^{3} \frac{1}{i} * X_i \right\} \quad \textbf{(56)}$$

Como puede apreciarse en **(56)**, este es un problema bi-objetivo. Si se evalúa el espacio de soluciones factibles del problema 1 en **(56)**, se obtienen los resultados de la tabla 6. El FP para este problema puede apreciarse en la figura 12.

Tabla 7.
Función F(X)aplicada a los estados qi, qi∈Q, del AFDM del ejemplo 1

Estado	**Asignación**			**Tiempos**			$\boldsymbol{F(X)}$	
q_i	M1	M2	M3	M1	M2	M3	$f_1(X)$	$f_2(X)$
q_0	P1	P2	P3	10	50	5	125	36.66
q_1	P1	3	P2	10	5	50	170	29.16
q_2	P2	P1	P3	50	10	5	85	56.66
q_3	P	P3	P1	50	5	10	90	55.83
q_4	P3	P1	P2	5	10	50	175	26.66
q_5	P3	P2	P1	5	50	10	135	33.33

Nota: En este caso en particular$F(X)$ recibe el orden en que se asignarán los trabajos en las tres máquinas y devuelve dos valores correspondientes a cada función objetivo (Nieto, 2011).

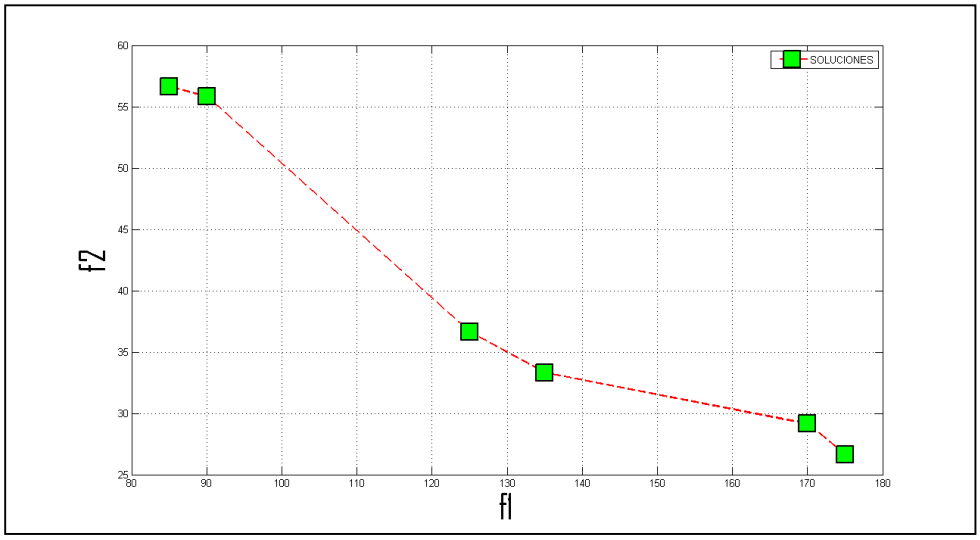


Figura 12. Conjunto de soluciones factibles para el problema del ejemplo 1
Nota: Conjunto de soluciones factibles para el problema del ejemplo 1 al utilizar la función 55, en verde se encuentran las soluciones (Nieto, 2011).

2.4. Computación flexible (Berlanga, 2010)

El término computación flexible (Soft computing) agrupa a un conjunto de metodologías y técnicas modernas de inteligencia artificial, comúnmente empleadas en minería de datos como redes neuronales (RNAs), Algoritmos genéticos, Razonamiento Probabilístico, los algoritmos evolutivos, lógica difusa, sistemas caóticos, redes de opinión, teoría de aprendizaje; no obstante, es necesario acentuar, que la computación flexible no es una mezcla de técnicas, sino una disciplina, en la cual cada componente contribuye con una metodología distinta para manejar problemas en su dominio de aplicación. Dadas las combinaciones entre ellas, se producen los denominados Sistemas inteligentes híbridos, entre los cuales se destacan los sistemas neuro-difusos, los sistemas difusos genéticos, los sistemas neuro-genéticos, los sistemas neuro-difusos-genéticos. Véase la figura13.

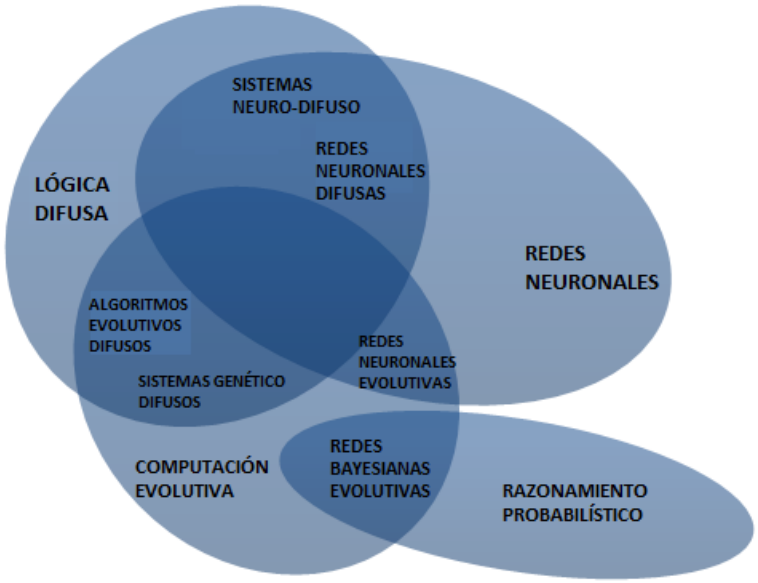


Figura 13. Hibridación Soft computing
Nota: Adaptado de (Cordón. o., Herrera. F, Magdalena. L., & Hoffmann. F, 2004).

Dentro de este enfoque Zadeh, (1994)establece que:

> *Básicamente, la computación flexible no es un cuerpo homogéneo de conceptos y técnicas. Es más bien una mezcla de distintos métodos que de una forma u otra cooperan desde sus fundamentos. En este sentido el principal objetivo de la computación flexible es aprovechar la tolerancia que conllevan la imprecisión y la incertidumbre, para conseguir manejabilidad, robustez y soluciones a bajo costo. Los principales ingredientes son la lógica difusa, la neuro computación y el razonamiento probabilístico, incluyendo este último a los algoritmos genéticos, las redes de creencia, los sistemas caóticos y algunas partes de la teoría de aprendizaje. En esta asociación de lógica difusa, la neuro computación y el razonamiento probabilístico, la lógica difusa se ocupa de la imprecisión y el razonamiento aproximado, la neuro computación del aprendizaje, y el razonamiento*

> *probabilístico de la incertidumbre y la propagación de las creencias.*(Berlanga, 2010,p.25)

Adicionalmente, en proceso de extracción de conocimiento en base de datos se puede enfocar y resolver como problemas de optimización y búsqueda. En esta sección se ampliará el concepto de cómputo flexible, y principalmente en dos de sus técnicas: los algoritmos evolutivos y lógica difusa, en que serán muy útil esta investigación para el desarrollo de nuevos algoritmos de minería de datos.

Existen diversas técnicas en el ámbito de la computación flexible como lo menciona Berlanga (2010).

- **Lógica difusa:** la lógica difusa trata con conjunto difusos (Zadeh, 1965) y conectores lógicos, de esta manera los conjuntos difusos permiten establecer límites flexibles entre los distintos niveles de significado, sin ignorar, ni enfatizar en exceso los elementos cercanos a las fronteras, de igual forma que ocurre en la percepción humana, como es saber, en todo proceso de extracción de conocimiento hay un componente de interacción humana y los conjuntos difusos permiten representar el conocimiento en forma lingüística, permitiendo conocimiento previo de forma fácil y proporcionar soluciones interpretables.

- **Redes neuronales artificiales:** Las redes neuronales son métodos predecibles no lineales que aprenden a través del entrenamiento y semejan la estructura de una red neuronal biológica; su principal utilidad es el aprendizaje automático, que por medio de varias iteraciones actualiza los parámetros hasta alcanzar un estado de equilibrio.

- **Algoritmos evolutivos:** La computación evolutiva está asociada al empleo de modelos evolutivos para el diseño e implementación de sistemas computacionales de alto nivel para la resolución de problemas de alta complejidad, como resultado, hay diferentes técnicas de

optimización y búsqueda sugeridas por los investigadores para realizar dicho trabajo. Según (Bäck, Fogel. D, & Michalewicz. Z, 1997)existen cuatro tipos básicos de algoritmos evolutivos: Los algoritmos genéticos, las estrategias de evolución, la programación evolutiva y la programación genética. En detalle, un algoritmo evolutivo se basa en mantener una solución de posibles soluciones del problema a resolver, realiza una serie de iteraciones genéticas de esa misma población y efectúa una selección para determinar qué soluciones permanecen en generaciones futuras y cuáles serán eliminadas.

- **Razonamiento probabilístico:** Es un tipo de modelo que se basa en la teoría de las probabilidades, por consecuente, es una de las técnicas de minería de datos que presenta que mayor dificultad al momento de cuantificar la incertidumbre. Históricamente, las redes bayesianas (Pearl J., 1988) reflejan una representación gráfica de dependencias para razonamiento probabilístico.

En el proceso de extracción de conocimiento en base de datos y, en particular, en el proceso de minería a de datos, existen muchas tareas que se pueden enfocar y resolver como problemas de optimización y búsqueda, dicho esto, podemos explotar las bondades de los algoritmos evolutivos, los cuales imitan los principios de la evolución natural para formar procedimiento de búsqueda y/o optimización global, que ofrecen grandes beneficios para crear nuestros sistemas artificiales si se emplean en el desarrollo de algoritmos de minería de datos, en el desarrollo de algoritmos de pre o pos procesamiento o como herramientas de para la optimización de los parámetros de otros algoritmos (Freitas, 2002).

Otro factor a considerar en la minería de datos, es la comprensibilidad de los resultados para el usuario, en este aspecto, la lógica difusa constituye una herramienta de representación del conocimiento que permite modelar incertidumbre e imprecisión de una forma sencilla y de fácil interpretación para el usuario.

En las últimas décadas, es notorio el interés de los científicos en datos sobre los algoritmos evolutivos de extracción de reglas difusas, debido alta potencialidad, como habíamos visto los algoritmos evolutivos tienen un carácter de búsqueda global que hacen de su implementación sea la más adecuadas para resolver problemas, tratan de forma adecuada las interacciones entre los atributos porque evalúan a la regla como un todo mediante la función de adaptación en lugar de añadir o eliminar una condición de una regla, como ocurre en los algoritmos de inducción de reglas y árboles de decisión.

En las siguientes secciones se describen los algoritmos evolutivos y la lógica difusa, que son objeto de estudio de en esta investigación.

2.4.1. Algoritmos evolutivos.

El enfoque de la computación evolutiva comprende un conjunto de algoritmos estocásticos de búsqueda basados en abstracciones del proceso de evolución de Darwin. Estos algoritmos se utilizan cada vez con mayor frecuencia para remplazar a los métodos clásicos en la resolución de problemas reales. Existen distintas tipologías de algoritmos evolutivos y presentan en común las siguientes características (Bäck et al., 1997)

Utilizan proceso de aprendizaje colectivo de una población de individuos. Generalmente cada individuo representa un punto dentro del espacio de búsqueda de todas las soluciones potenciales para un problema dado, es decir, codifica una solución candidata. Los individuos pueden incorporar adicionalmente otra información, como pueden ser los parámetros de la estrategia del algoritmo evolutivo. En el área de la computación evolutiva a la solución codificada se le denomina genotipo y a la solución decodificada (lo que realmente representa cada individuo en el contexto del problema) se le denomina fenotipo.

Los descendientes de los individuos se generan mediante procesos no determinísticos que tratan de modelar los procesos de mutación y cruce. La mutación corresponde a una auto replicación errónea de los individuos genético entre dos o más individuos ya existentes. Ambos operadores son estocásticos, se aplican con probabilidades definidas por el usuario.

Normalmente se establece una probabilidad de mutación muy inferior a la probabilidad de cruce, ya que una probabilidad de mutación muy elevada convertiría el proceso de búsqueda evolutivo en un proceso de búsqueda aleatoria. No obstante, la mutación es necesaria para la incrementar la diversidad genética de los individuos dentro de la población y para alcanzar valores de genes que no estén presentes en la población y que de otra forma serian inalcanzables, puesto que el operador de cruce sólo intercambia genes (ya existentes) entre individuos.

Se asigna una medida de calidad (denominada medida de adaptación o fitness) a cada individuo mediante el proceso de evaluación. El operador de selección actúa en base a esta medida y favorece, en el proceso de reproducción, a individuos mejores respecto a aquellos con peor valor de la función de adaptación.

A continuación, se enuncian los cuatros modelos computacionales dentro de los algoritmos evolutivos:

- Algoritmos Genéticos (Goldberg. D. E, 1989), (Holland, 1975), que modelan la evolución genética, por lo que las características de los individuos se expresan como genotipos.
- Estrategias de evolución (Schwefel H. P., 1995), que se orienta hacia la modelación de los parámetros estratégicos que controlan la variación en la evolución, es decir. La evolución de la evolución.
- Programación Evolutiva (Fogel D. B, 1988), derivada de la simulación de comportamiento adaptativo de la evolución.

- Programación Genética (Koza J. R., 1992) , (Koza J. R., 1994), basadas en algoritmos genéticos, pero los individuos son programas que se representan mediante árboles.

ALGORITMO 1: Pseudocódigo de un Algoritmo Evolutivo.

```
1. P ← GenerarPoblaciónInicial();
2. Evaluar(P);
3. mientras !CondiciónParada() hacer
4.       P' ←SeleccionarPadres(P);
5.       P' ←OperadoresDeVariación(P');
6.       Evaluar(P');
7.       P ← SeleccionarNuevaPoblación(P, P');
8. fin mientras
9. Resultado: La mejor solución encontrada
```

Figura 14. Pseudocódigo algoritmo evolutivo
Nota: Adaptado de(Ruiz, 2013).

El funcionamiento de cualquier algoritmo evolutivo se puede describir de la siguiente forma (ver figura 14), se mantiene una solución de posibles soluciones para el problema, se realizan modificaciones sobre las mismas y se seleccionan en función de una medida de adaptación del individuo al entorno, aquellas que se mantendrán en generaciones futuras y las que serán eliminadas. La población evoluciona a través de las mejores regiones del espacio de búsqueda mediante los procesos de modificación y selección. Las modificaciones sobre la población permiten mezclar información de los padres que debe pasar a los descendientes (operador de cruce) o introducir innovación dentro de la población (operador de mutación).

Una característica importante de los algoritmos evolutivos es que realizan un proceso de búsqueda global. El hecho de que trabajen con una población de soluciones candidatas más que

una solución individual, junto con el uso de operadores estocásticos, reduce la probabilidad de caer en un óptimo local e incrementa la probabilidad de encontrar el máximo global.

Otra característica de los algoritmos evolutivos es el carácter de búsqueda global que hace que sean especialmente adecuados para resolver problemas presentes en las distintas etapas del proceso de descubrimiento de conocimiento (Freitas, 2002). Por ejemplo, en proceso de extracción de reglas, los algoritmos evolutivos tratan de forma adecuada las interacciones entre atributos porque evalúan una regla como un todo mediante la función de adaptación en lugar de evaluar el impacto de añadir y/o eliminar una condición de una regla como ocurre en los procesos de búsqueda local incluidos en la mayoría de los algoritmos de inducción de reglas y árboles de decisión. En(Bäck et al., 1997) se puede encontrar de forma detallada los distintos tipos de algoritmos evolutivos.

Entre las distintas clases de algoritmos evolutivos, los algoritmos genéticos (AGs) y la programación genética (PG) son los más utilizados para el descubrimiento de reglas. Estas dos clases de algoritmos difieren fundamentalmente en la representación de los individuos. En los AGs, los individuos se representan como una cadena lineal de condiciones, y en el caso de reglas cada condición suele ser una pareja atributo valor, mientras que en PG un individuo suele representarse mediante un árbol, y en este caso particular los nodos hoja o terminales son condiciones de reglas y/o valores de atributos, y los nodos internos representan las funciones.

En este trabajo se abordará la extracción de reglas de clasificación utilizando AGs.

2.4.2. Algoritmos genéticos (AG).

El proceso de extracción de conocimiento en base de datos se puede enfocar y resolver como problemas de optimización y búsqueda, de lo anterior, los problemas de optimización se pueden clasificar de acuerdo a los algoritmos que se apliquen para su resolución. El enfoque completo lo muestra la figura 15.

Generalmente, los algoritmos de clasificación se dividen en dos categorías: **Deterministas y estocásticos.** Los algoritmos determinísticos no cambian su recorrido en la trayectoria ejemplo Hill-Climbing, a diferencia de los estocásticos tiene un componente aleatoriedad por ejemplo algoritmos genéticos, la solución de la población difiere en cada ejecución. No obstante, se empleanhibridaciones de deterministas y algoritmos estocásticos. Por ejemplo, el Hill-Climbing con reinicio aleatorio. Este método utiliza el algoritmo determinista, pero comienza con diferentes puntos iniciales. Su implementación ha sido probada en distintas áreas de aplicación tales como redes y comunicación, robótica minería de datos y análisis de datos, comportamientos en evolución, por ejemplo, para agentes o jugadores de juegos, optimización combinatoria (Weise, 2008).

Pasando al otro enfoque, el estocástico está determinado por los algoritmos: heurísticos y metaheurísticos. Siendo éste materia de estudio en esta investigación, específicamente los algoritmos meta heurísticos.

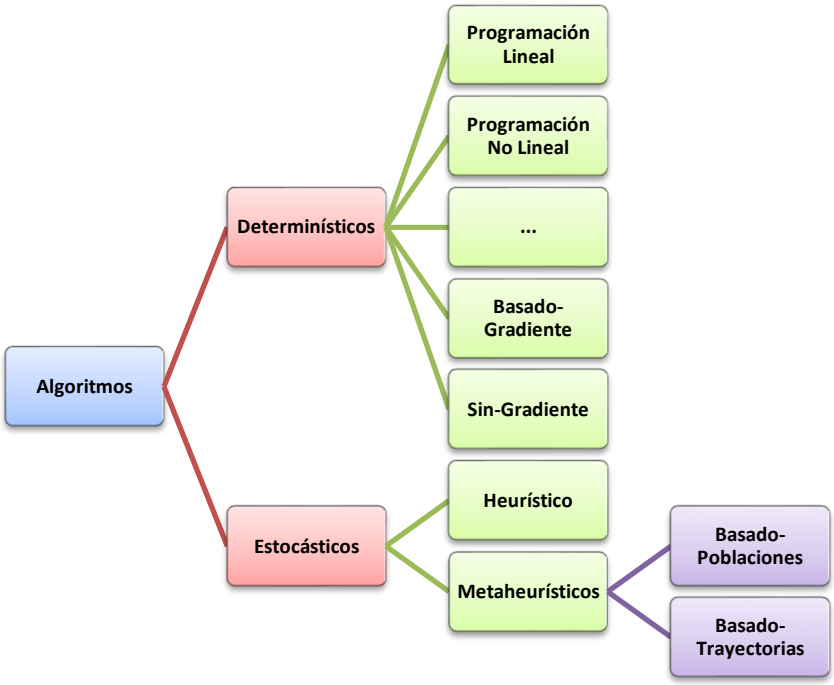


Figura 15. Clasificación de los algoritmos
Nota: Adaptado de (Ruiz, 2013).

Según, con el párrafo anterior, Los AGs son algoritmos estocásticos de optimización y búsqueda inspirados en los procesos de evolución natural, fueron definidos inicialmente por Holland, (1975) y han sido estudiados por otros autores Bäck, Fogel y Michalewicz (Bäck et al., 1997), Goldberg.(1989). Este último autor establece que:

"Los algoritmos de búsqueda basados en los mecanismos de selección natural y genética natural. Combinan la supervivencia de los más compatibles entre las estructuras de cadena, con una estructura de información ya aleatorizada, intercambiada para construir un algoritmo de búsqueda con algunas de las capacidades de innovación de la búsqueda humana" (Genetics Algorithms in Search, optimization and machine learning. Adinson Wesley, 1989, p.1).

Funcionamiento del algoritmo genético.

El funcionamiento de un algoritmo genético es el siguiente: (Michalewicz Z., 1996).Véase Figura 16 pseudocódigos de un algoritmo genético.

```
Procedimiento Algoritmo Genético
  EMPEZAR
        t = 0;
        inicializar Pob(t);
        evaluar Pob(t);
        MIENTRAS NO (condición de parada) HACER
          EMPEZAR
                t = t + 1;
                seleccionar Pob'(t) a partir de Pob(t − 1);
                cruzar y mutar Pob'(t);
                Pob(t) = Pob'(t);
                evaluar Pob(t);
          FIN
  FIN
```

Figura 16. Pseudocódigo de un algoritmo genético
Tomado de (Michalewicz Z., 1996).

El sistema parte de una población inicial de individuos que codifican soluciones candidatas al problema propuesto mediante alguna representación genética, esta población de individuos (denominados cromosomas) evolucionan en el tiempo a través de un proceso de competición y variación controlada, en donde cada cromosoma de la población tiene asociada una medida de adaptación para determinar que cromosomas serán seleccionados para formar parte de la nueva población en el proceso de competición. En este sentido la nueva población se creara utilizando operadores genéticos de cruce y mutación, y el ciclo evolutivo continua hasta que se verifique una determinada condición de parada, como puede ser por ejemplo, que se ha realizado un determinado número máximo de generaciones de individuos, que la población haya evolucionado durante un número máximo de generaciones, que se haya alcanzado una solución con un determinado valor

de la función de adaptación (o parte de ella), que la población no evolucione por no generarse individuos nuevos durante un determinado número de generaciones.

Para la comparación de algoritmos Holland distingue a los AGs de otros algoritmos evolutivos por tres características (Holland, 1975): El esquema de codificación binario, el método de selección (proporcional a la función de adaptación) y el método para producir variaciones en la población, (el operador de cruce), esta última característica es el componente diferenciador entre los AGs y los algoritmos evolutivos. Mientras que, por otro lado, se utilizan métodos alternativos de selección y se adoptan esquemas de codificación adaptados a los problemas a resolver y menos restrictivo que el esquema de codificación binario.

Por otra parte, la aplicación de AGs para resolver un problema debe determinar:

- Una representación genética de las soluciones del problema.
- Una forma de crear una solución inicial de soluciones.
- Una función de evaluación que proporcione un valor de adaptación de cada cromosoma.
- Operadores que modifiquen la composición genética de la descendencia durante la reproducción
- Valores para los parámetros utilizados (como tamaño de la población, probabilidades de aplicación de los operadores genéticos).

A continuación, se detallan los aspectos relacionados como: la representación de las soluciones, el mecanismo de selección y los operadores genéticos de cruce, mutación y permutación.

2.4.2.1. Representación de las soluciones.

Hay diferentes representaciones disponibles para el cromosoma y la selección de la representación adecuada es específica del problema. La buena representación es lo que hace que

el espacio de búsqueda sea más pequeño y, por lo tanto, una búsqueda más fácil. Existen distintos esquemas generales de codificación se destacan los siguientes:

Codificación binaria: Se basa en la representación de cromosomas como cadenas de bits, es decir, ceros y unos, fue la primera codificación que se utilizó (Goldberg. D. E, 1989), (Holland, 1975).

Codificación basada en orden: se basa en la representación de cromosomas en los que las soluciones son permutaciones de un conjunto de elementos determinados (Goldberg. D. E, 1989), (Michalewicz Z., 1996), (Michalewicz Z., 1996), este esquema es útil para problemas de optimización combinatoria.

Codificación real: Se basa en la representación de cromosomas que incluyen variables continuas (Herrera F., Lozano, & L. Verdegay J., 1998), en este esquema de representación cada variable del problema se asocia a un único gen que toma un valor real dentro del intervalo especificado, por lo que no existen diferencias entre el genotipo (la codificación empleada) y el fenotipo (la propia solución codificada).

Por ejemplo, si vamos a codificar el número 7 en binario, podría tener el siguiente aspecto:

3	2	1	0
0	1	1	1

Cada parte del cromosoma anterior se llama gen. Cada gen tiene dos propiedades. El primero es su valor (alelo) y el segundo es la ubicación (locus) dentro del cromosoma, que es el número por encima de su valor.

Cada cromosoma tiene dos representaciones.

- **genotipo**: El conjunto de genes que representan el cromosoma.
- **fenotipo**: La representación física actual del cromosoma.

En el ejemplo anterior, el binario de 0111 es el genotipo y 7 es la representación del fenotipo.

Después de representar a cada cromosoma la forma correcta de servir para buscar en el espacio, a continuación, se calcula el valor de aptitud de cada individuo. Supongamos que la función de aptitud utilizada en nuestro ejemplo es:

f (x) = 2x + 2 Donde x es el valor del cromosoma

Entonces el valor de aptitud del cromosoma anterior es:

f (7) = 2 (7) + 2 = 16

El proceso de cálculo del valor de aptitud física de un cromosoma se denomina evaluación.

2.4.2.2. El mecanismo de selección.

Es el encargado de seleccionar un número de individuos de la población en el grupo de apareamiento. Según el valor de aptitud física calculado previamente, se seleccionan los mejores individuos en función de un umbral. Después de ese paso, terminaremos seleccionando un subconjunto de la población en el grupo de apareamiento.

Existen diferentes formas de llevar a cabo la selección (Bäck & Schwefel H., 1991) , como la selección proporcional (Holland, 1975), ranking (Baker J. E., 1987), o por torneo (Brad & Goldberg, 1995) entre otras. El mecanismo de selección puede ser complementado por el modelo de selección elitista, basado en mantener un número determinado de individuos mejor adaptados de la población anterior en la nueva población (la obtenida después de llevar a cabo el proceso de selección y de aplicar operadores de cruce y mutación) (Goldberg. D. E, 1989), (Michalewicz Z., 1996) .

2.4.2.3. Modelos de la población (Berlanga, 2010).

Modelo selectivo se ve afectado por la forma en que los cromosomas de la población son reemplazados por los nuevos descendientes. Por ello, se pueden utilizar métodos de remplazamiento aleatorios, o determinísticos.

- El modelo elitista. Consiste en utilizar una población intermedia con todos los padres y todos sus descendientes para seleccionar los mejores, también se puede optar por no remplazar al mejor cromosoma de la población para no perder la mejor solución encontrada.
- Modelo generacional. Una vez aplicados los operadores de cruce y mutación en cada iteración se crea una población completa con nuevos individuos, es decir, después de cada generación se reemplaza la población completa por sus descendientes, obteniendo de esta forma la siguiente generación (Holland . H. J., 1975).
- Modelo estacionario. En este modelo se modifica solo una parte de la población, el procedimiento requiere el remplazar a uno de los padres por alguno de sus descendientes (Whitley L. D. y Kauth J., 1988).

Operadores genéticos (**Weise, 2008)**

Sobre la base de los individuos seleccionados en el grupo de apareamiento, los padres son seleccionados para el apareamiento, la selección de cada uno de los dos padres puede ser seleccionando los padres de forma secuencial (1-2, 3-4, etc.). Otra forma es la selección aleatoria de los padres.

Para el caso de una generación a la siguiente se aplican una serie de operadores genéticos, los más empleados son los operadores de selección, cruce o recombinación, copia y mutación.

El operador de cruce

Este operador constituye un mecanismo para compartir información entre cromosomas. Por medio de este operador se combinan las características de dos cromosomas padres adaptados. Para ello la descendencia toma la mitad de sus genes de un padre y la otra mitad del otro padre, teniendo en cuenta que la cantidad de genes transportados de cada padre es aleatoria y para cada dos padres,

el cruce se lleva a cabo seleccionando un punto aleatorio en el cromosoma e intercambiando genes antes y después de ese punto, por lo tanto, el operador se denomina cruce de un solo punto. Todo esto indica que para la transición entre generaciones el cruce es importante y sin él, la descendencia será idéntica a su padre.

Existe muchos algoritmos de cruce, sin embargo, lo más empleados son los siguientes:

- Cruce de un punto
- Cruce de dos puntos
- Cruce uniforme

El cruce de un punto: Una vez seleccionados dos individuos se cortan sus cromosomas por un punto seleccionado aleatoriamente, para generar dos segmentos diferenciados en cada uno de ellos: La cabeza y la cola. Se intercambian las colas entre los dos individuos para generar los nuevos descendientes, de esta manera ambos descendientes heredan información genética de sus padres. Véase la figura17.

El cruce de dos puntos: Es el mismo proceso del cruce de 1 punto, se realizan dos cortes en los cromosomas de los padres, debe tenerse en cuenta que los puntos de corte no coincidan con el extremo de los cromosomas para garantizar que se originen tres segmentos. Para generar la descendencia se escoge el segmento central de uno de los padres y los segmentos laterales del otro padre. Véase en la figura 17 un ejemplo de cruce en dos puntos. Practicando la misma filosofía se pueden añadir más puntos, generando algoritmos de cruce multipunto, sin embargo estudios refutan esta técnica (De Jong. K.A, 1975), quien afirma que el hecho de añadir un mayor número de puntos de cruce reduce el rendimiento del algoritmo genético, sostiene que es más fácil que los segmentos originados sean fácilmente corrompibles, es decir que por separado pueden perder las

características que poseían en conjunto, sin embargo, asevera que al añadir más puntos de cruce se consigue que el espacio de búsqueda del problema sea explorado en profundidad.

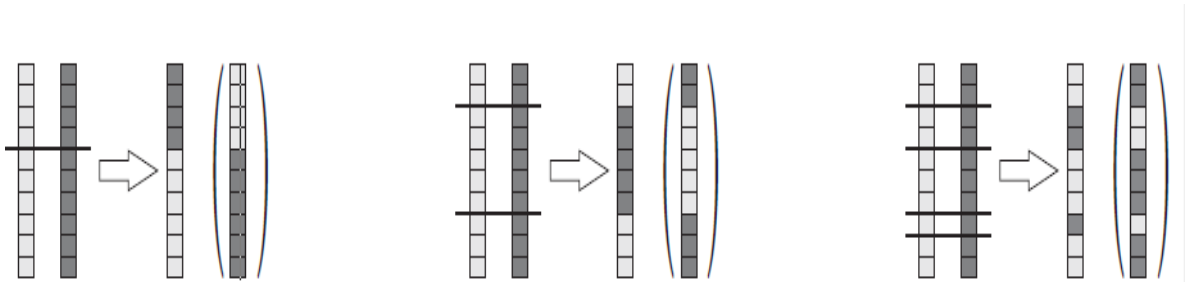


Figura 17. Operador de cruce
Nota. Los operadores mayormente utilizados son cruce de un punto, cruce de dos puntos, cruce multipuntos. Adaptado de (Weise, 2008)

El cruce uniforme: Cada gen de la descendencia tiene las mismas posibilidades de pertenecer a uno u otro progenitor, la técnica implica la generación de una máscara de cruce con valores binarios. Si en la posición de la máscara hay un 1, el gen situado en esa posición de los descendientes se copia del primer progenitor, si, por el contrario, hay un 0 el gen se copia del segundo progenitor. Para el segundo descendiente se intercambian los papeles de los padres, o se intercambian la interpretación de los unos y los ceros de la máscara de cruce. Véase en la figura 18como se puede apreciar la máscara de cruce no permanece fija durante el proceso evolutivo, se evidencia el componente aleatorio para cada cruce.

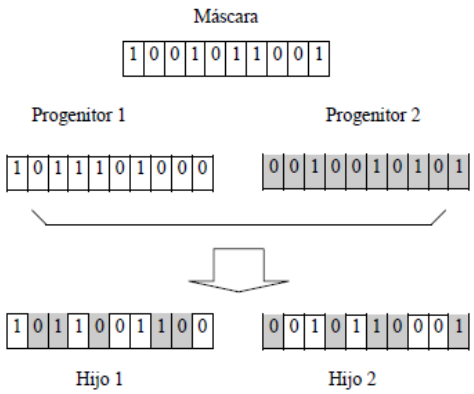


Figura 18. Ejemplo de cruce uniforme
Nota: Adaptado de(Weise, 2008)

Para genotipos con codificación compuesto por valores enteros o reales los operadores de cruce que se emplean son los siguientes:

- Media: El gen de la descendencia toma el valor medio de los padres, la única desventaja es que se genera un descendiente en el cruce de dos progenitores.
- Media geométrica: Cada gen de la descendencia toma como valor la raíz cuadrada del producto de los genes de los padres, el problema es la asignación del signo al descendiente puesto que los padres pueden ser de diferentes signos.
- Extensión: Se toma la diferencia existente entre los genes situados en las mismas posiciones de los padres y se suma el valor más alto o se resta el valor más bajo. Solventa el problema de generar un único descendiente.
- Copia: Consiste en la copia de un individuo en la nueva generación, el porcentaje de copias de una generación a la siguiente es relativamente reducido, caso contrario se corre el riesgo de una convergencia prematura de la población hacia ese individuo. Generalmente se seleccionan dos individuos para el cruce y si este finalmente no tiene lugar, se insertan en la siguiente generación los individuos seleccionados.

Mutación

La mutación es un método importante para preservar la diversidad de los candidatos. Consiste en alterar arbitrariamente uno o más genes del cromosoma seleccionado para aumentar la diversidad de la población, generalmente la alteración aleatoria de uno de los componentes de código genético de un individuo suele conllevar a un salto a otra zona del espacio de búsqueda que puede resultar más prometedora. El operador de mutación que generalmente se utiliza en los AGs es el binario que consiste simplemente en negar un bit; La Figura 19, muestra la variante más general de esta forma de mutación. También es posible realizar la mutación con codificación real,

de forma que el nuevo valor del gen mutado se escoja aleatoriamente dentro del intervalo de definición asociado. Con las siguientes opciones:

- Incrementar o decrementar a un gen una pequeña cantidad generada aleatoriamente.
- Multiplicar un gen por un valor aleatorio próximo a 1.

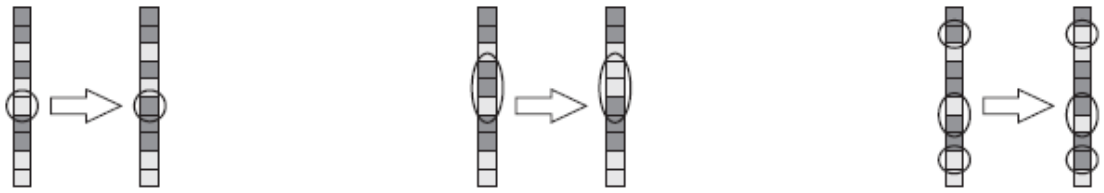


Figura 19. Mutación que altera el valor de los cromosomas en serie
Nota; El primer esquema representa la mutación de un solo gen, el segundo y el tercer esquema representa la mutación multigénica Adaptado de (Weise, 2008).

Permutación: Reproducción

La operación de permutación es un método de mutación alternativo en el que los alelos de dos genes se intercambian como se muestra en la figura 20. Esto, por supuesto, solo tiene sentido si todos los genes tienen tipos de datos similares. La permutación es, por ejemplo, útil cuando se resuelven problemas que implican encontrar una secuencia óptima de artículos, como el problema del vendedor ambulante (Kangshun. Li, Yuanxiang. Li, Mo, & Chen, 2005), (E. L. Lawler, Jan Karel Lenstra, A. H. G. Rinnooy Kan, 1985). Aquí, un genotipo g podría codificar la secuencia en la que se visitan las ciudades. Intercambiando dos alelos entonces es igual a cambiar dos ciudades en la ruta.

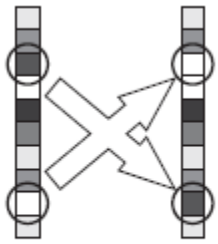


Figura 20. Permutación aplicada a un cromosoma
Nota: Adatado de (Weise, 2008)

Algoritmos de reemplazo

Para insertar un nuevo individuo debe eliminarse previamente otro de la población. Para ello existen diferentes métodos de reemplazo:

- Aleatorio: El nuevo individuo se inserta en un lugar cualquiera de la población.
- Remplazo de padres: se obtiene espacio para la nueva descendencia liberando el espacio ocupado por los padres
- Remplazo de similares: Una vez obtenido el ajuste de la descendencia se selecciona un grupo de individuos (entre seis y diez) de la población con un ajuste similar, y se remplazan aleatoriamente los que sean necesarios.
- Remplazo de los peores: De entre un porcentaje de los peores individuos de la población se seleccionan aleatoriamente los necesarios para dejar sitio a la descendencia.

2.4.2.4. Evaluación.

Para validar el correcto funcionamiento de un AG se debe poseer un método que mida si los individuos de la población representan o no buenas soluciones al problema planteado, de esto se encarga la función de evaluación (fitness, ajuste) que establece una medida numérica de la bondad de la solución.

Darwin califica la función de ajuste como la habilidad que tiene un individuo de competir en entorno por los recursos disponibles Goldberg., (1989) describió la función de ajuste como "una medida de beneficio, utilidad o bondad que queremos maximizar", Koza,(1992) proporciona tres modelos para valorar esta medida de ajuste:

Estandarizado. Esta medida de ajuste $s(i,t)$ mide la bondad de un individuo i en la generación t, de tal forma que los valores próximos a 0 indica un buen valor de ajuste y los valores lejanos de 0, significa un mal individuo, es decir, malas soluciones. Por tanto, En una generación t un individuo i será peor j, si se cumple $s(i,t)>s(j,t)$Esta medida es muy utilizada en problemas donde el nivel de ajuste se realiza penalizaciones, por ejemplo, error cuadrático medio, error de inducción de fórmulas, número de ejecuciones necesarias para encontrar la solución.

Ajustado. Esta medida se obtiene a partir del ajuste estandarizado, y se obtiene de la siguiente expresión:

$$\alpha(i,t) = \frac{1}{1+s(i,t)} \qquad \textbf{(57)}$$

Con el valor obtenido de la medición la bondad se cuantifica entre 0 y 1, siendo 1 el valor correspondiente al mejor individuo.

Normalizado. Esta medida corresponde a un valor de ajuste comparativo del individuo con toda la población. Se obtiene de la siguiente expresión:

$$n(i,t) = \frac{\alpha(i,t)}{\sum_{k=1}^{M} a\,(k,t)} \qquad \textbf{(58)}$$

El valor obtenido de la medición está comprendido entre 0 y 1. Este tipo de ajuste indica el tipo de bondad de la población. Por tanto, un valor cercano a 1 no indica que el individuo representa

una solución buena al problema, sino que el individuo representa una solución destacada y notablemente mejor que el resto de la población. La justificación de emplear este concepto de evaluación consiste en crear una medida de ajuste para cada individuo de la población y es específico del problema en el que se aplica el algoritmo genético.

2.4.2.5. Parámetros.

Los principales parámetros que se pueden configurar para variar la ejecución son:

Tamaño de la población. Indica el número de individuos que va a tener la población, en general este parámetro se ajusta de forma proporcional a la complejidad del problema, cuando el tamaño de la población toma valores altos habrá más opciones de conseguir mejores resultados en un menor número de generaciones. Sin embrago un tamaño alto puede no ser siempre la mejor solución, es posible tomar un tamaño menor y confiar en la evolución durante mayor número de generaciones (Fuchs, 1999), (Gathercole & Ross, 1997).

La tasa de cruces. El porcentaje de individuos de la siguiente generación creados a partir de la de cruces de individuos de la anterior, suele generalmente superar el 90%.

La probabilidad de mutación. Suele ser muy baja menor de 0.05

Los algoritmos de cruce, selección y mutación.

El criterio de parada del algoritmo

- Número máximo de generaciones.
- Haber alcanzado un número de ajuste aceptable.
- Convergencia de la población.

2.4.3. Metaheurística evolutiva con autómatas.

Dentro de este marco se especificarán los métodos de optimización, tanto heurísticos como metaheurísticos y sus definiciones.

Heurística.

Desde hace mucho tiempo se emplean diversas estrategias para tratar de resolver problemas de interés, a pesar de ello, no se conoce un algoritmo exacto con complejidad polinómica que encuentre la solución óptima, más aun cuando la cardinalidad del espacio de búsqueda suele ser enorme, lo cual impide el nivel de acierto de los algoritmos exactos, dado que la cantidad de tiempo que requieren para encontrar una solución es demasiado alto, por estos motivos surge la necesidad de implementar métodos heurísticos que permitan obtener una buena solución de calidad en un tiempo razonable.

Actualmente existen varias definiciones de heurística (Melián, Moreno Pérez, Marcos, & Vega, 2003):

Definición 1: Procedimientos simples que empleando conocimiento a cerca de un problema y de las técnicas aplicables tratan de aportar una buena solución (no necesariamente la óptima) usando una cantidad de recursos computacionales en un tiempo razonable.

Definición 2: En optimización, además de las condiciones que debe cumplir las soluciones factibles del problema de interés, se requiere que la solución sea la óptima, según los criterios de comparación entre ellas.

Definición 3: Procedimiento que produce un alto grado de confianza, que encuentra soluciones de alta calidad con un coste computacional razonable, aunque no garantice su óptimal, o su factibilidad. E inclusive no llegue a determinar lo cerca que pueda estar de la solución.

Por otro lado, los métodos heurísticos tienen como desventaja, la incapacidad para escapar de los óptimos locales, esto se bebe a que carecen de, mecanismos que permitan perseguir la búsqueda

de la solución óptima. Para solventar este problema, se introducen nuevos algoritmos de búsqueda con un carácter inteligente denominados Meta heurísticas, estas son estrategias de alto nivel que ayudan a los métodos heurísticos, y evitan que queden atrapados en óptimos locales.

Metaheurística

El término metaheurística fue acuñado por Glover en el año 1986, proviene de la composición de dos palabras de origen griego meta y heurística. Meta que significa "más allá" o "a un nivel superior". Actualmente existen varias definiciones de meta heurística una de las definiciones más descriptiva es la presentada por (Melián et al., 2003):

Definición: Son estrategias inteligentes para diseñar o mejorar procedimientos heurísticos para la resolución de problemas con un alto rendimiento.

A partir de entonces han surgido multitud de propuestas en lo referente al diseño y procedimientos para resolver problemas de interés general.

Tipos de metaheurísticas.

En función del tipo de procedimiento a los que se refiere, existen varios enfoques para las metaheurísticas.

- **Meta heurística de relajación:** Procedimientos de resolución de problemas que utilizan relajaciones del modelo original, es decir modificaciones en el modelo para hacerlo más fácil de resolver.
- **Meta heurística constructiva:** procedimientos que tratan de la obtención de una solución a partir del análisis y selección paulatina de los componentes que lo forman.
- **Meta heurística de búsqueda:** Guían a los procedimientos que usan transformaciones o movimientos para recorrer el espacio de soluciones alternativas y explotar las estructuras de entornos asociadas.

- **Meta heurísticas evolutivas:** Están enfocadas a los procedimientos basados en conjunto de soluciones que evolucionan sobre el espacio de soluciones.

*2.4.3.1. **clasificación de las metaheurística.***

Los problemas multi-objetivos se pueden abordar por distintas técnicas de optimización. A continuación, una descripción de la clasificación de los algoritmos y metaheurística comúnmente empleadas (véase Figura 21).

Clasificación de las metaheurística según el método de operación.

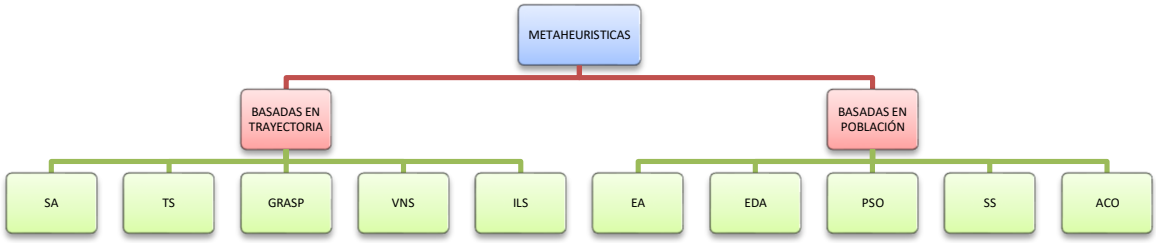


Figura 21. Clasificación de las metaheurística
Nota Adaptado de (Ruiz, 2013).

Metaheurísticas basadas en trayectoria.

Como complemento se enunciarán algunas metaheurística basadas en trayectoria, el objeto de esta metaheurística es que parte de una solución, y mediante la exploración del vecindario, van actualizando la solución, formando una trayectoria(Ruiz, 2013).

- Enfriamiento simulado (SA).
- Búsqueda Tabú (TS).
- Grasp
- Búsqueda con vecindario Variable (VNS).

- Búsqueda local iterada (ILS).

Metaheurísticas basadas en población.

El objeto de esta metaheurística como su nombre lo indica, trabaja con un conjunto de soluciones denominadas población en cada iteración. Como complemento se enunciarán algunas metaheurísticas basadas en población.

- Algoritmos evolutivos (AE).
- Algoritmo de estimación de la distribución (EDA).
- La búsqueda dispersa (SS).
- Algoritmos de optimización basados en colonia de hormigas (ACO).
- Algoritmos de optimización basados en cúmulos de partículas (PSO).

De acuerdo, con el seguimiento de los algoritmos expuestos, se hará énfasis en el análisis del ciclo de un algoritmo evolutivo por ser el objeto de estudio en esta investigación y algunos algoritmos referenciados en la literatura, adicionalmente, también enunciare aquellos algoritmos que se han fusionado con autómatas y, además, se han utilizado con éxito en diversos problemas y aplicaciones reales. Véase figura 22.

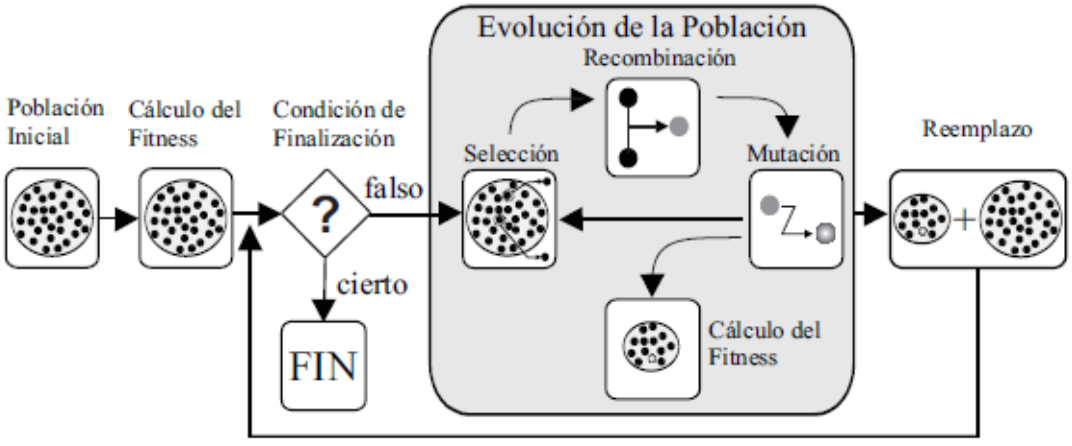


Figura 22. Ciclo de algoritmo evolutivo
Nota: adaptado de (Hernández Riaño, López Pereira, & Hernández Riaño, 2013b).

Hernández Riaño et al.,(2013b) explica el proceso que realiza el ciclo de un algoritmo evolutivo:

> A una población de "n" individuos se aplica una evaluación para determinar los mejores individuos en función de la aptitud o fitness. De estos mejores individuos se aplica una selección para determinar los individuos que serán elegidos como padres, haciendo que se reproduzcan con la ayuda de los operadores genéticos. Estos causan una serie de transformaciones de alteración y recombinación en los individuos, para generar los descendientes. Este ciclo se repite hasta conseguir hacer evolucionar la población, (p. 68).

Algoritmo evolutivo ébola. Es una metaheurística evolutiva inspirada en la propagación del virus en las células del cuerpo humano. Las soluciones llamadas, células infectadas, se "replican" en el espacio de búsqueda guiadas por las células inicialmente infectadas, luego las células liberan los nuevos virus que procederán a infectar a nuevas células y a los tejidos circundantes, las cuales representan nuevas soluciones del espacio factible, esto quiere decir, que cada célula infectada representa una solución factible. Para generar el tejido cercano a las células, se construye una matriz que contiene en cada fila rutas que difieren de la nueva célula, en un solo nodo (Hernández Riaño et al., 2013b).

Esta metaheurística aporta una estrategia para afrontar la resolución de un problema realizando una búsqueda sobre un espacio cuyos elementos representan las soluciones candidatas alternativas. Por este motivo se adoptará ésta metaheurística en el desarrollo de esta investigación.

En Yañez.,(2009) muestra un listado de los siguientes algoritmos evolutivos.

VEGA (Vector Evaluated Genetic Algorithm) propuesto por Scheffer, basado en el algoritmo génesis, modificando la forma de realizar la selección a fin de poner manejar varias funciones objetivas.

MOGA (Multi Objetive Genetic Algorithm) en este esquema aplica jerarquía a los individuos basándose en la dominancia de Pareto. Individuo (s) que no son dominados por otros se le asigna la mejor jerarquización y para los demás el valor de su jerarquía es igual al número de soluciones que lo dominan.

PAES (Pareto Archived Evolution Strategy) es una estrategia evolutiva del tipo (1+1) hace uso de un archivo histórico que almacena las soluciones no dominadas encontradas, cada nuevo individuo es comparado con el archivo histórico.

PESA (Pareto Envelope based selection Algorithm) esquema utiliza dos poblaciones, la interna de tamaño reducido y la externa de mayor tamaño, utiliza el mismo mecanismo de malla adaptativa que el PAES, el mecanismo de selección se basa en la métrica de densidad usada por la malla adaptativa y el archivo externo determina la selección realizada por el método. Implementa una malla adaptativa para mantener la diversidad de las soluciones en el archivo y divide el espacio de las funciones objetivos para agrupar las soluciones encontradas.

2.4.4. Algoritmos evolutivos con autómatas.

En esta sección se describe algunos algoritmos evolutivos multi-objetivo con autómatas finitos en procesos de optimización combinatoria, Niño.,(2012).

MIDA: Metaheurística de intercambio determinista sobre autómatas, (Niño& Ardila., 2009) en su investigación Proponen un Autómata Finito Determinista de Intercambio (AFD-I) que permite representar el espacio de soluciones factibles para un problema combinatorio , pero solo soluciona problemas combinatorios mono-objetivo, lo cual no permite resolver problemas más complejos. MIDA es contrastada con Metaheurística de gran impacto a nivel mundial tales como Algoritmos

basados en Colonias de Hormigas, Técnicas Evolutivas y estrategias de Búsqueda Local, todas ellas referenciadas de publicaciones en Journals Internacionales ISI (Indexadas por la ISI Web of Knowledge).

MODS: Metaheuristic Of Deterministic Swapping, (Niño et al., 2011) es una estrategia de búsqueda local que explora el espacio de solución factible de un problema combinatorio admitido en una estructura de datos denominada Autómatas finitos deterministas de múltiples objetivos MDFA (Child, Ardila, Donoso & Jabba, 2010).

SAMODS: **Simulated Annealing Metaheuristic Of Deterministic Swapping** desarrollada por Niño (2012) es una estrategia de búsqueda local híbrida basada en la teoría MODS y el algoritmo de recocido simulado para la optimización multi-objetivo de problemas combinatorios. Su propuesta consiste en optimizar un problema combinatorio utilizando una dirección de búsqueda y una mejora de ángulo removiendo las soluciones dominadas cuando encuentra soluciones no dominadas, además, emplea criterio de elitismo si la nueva solución encontrada domina al menos una solución del elemento. Sin embargo, cuando una solución dominada es encontrada SAMODS trata de mejorarla usando un mecanismo que prevea esta condición, la manera de hacerlo es aceptando las soluciones dominadas como buenas soluciones. Las soluciones dominadas son aceptadas bajo Boltzmann Distribution Probability assigning, es decir asignarles pesos a los objetivos del problema. De esta manera evita los óptimos locales.

SAGAMODS: Simulated Annealing, Genetic Algorithm & Metaheuristic Of Deterministic Swappingel recocido simulado, el algoritmo genético y la metaheurística del intercambio determinístico (Niño, 2012)es una estrategia de búsqueda híbrida basada en la teoría de los autómatas, los algoritmos de recocido y genéticos simulados, su estructura se basa en el esquema de SAMODS, aunque, Samods puede evitar los óptimos locales, puede llevar mucho tiempo aceptar soluciones dominadas para encontrar soluciones no dominadas. Por lo tanto, Sagamods

propone un paso de cruce antes de ejecutar Samods. Debido a esto, Sagamods es compatible con Samods para explorar regiones distantes del espacio de la solución.

EMODS: Evolutionary Metaheuristic Of Deterministic Swapping permite la optimización multi-objetivo de problemas combinatorios. Su estructura se basa en el esquema de MODS, por lo tanto, su método es el mismo: Crear soluciones iníciales, mejorar las soluciones (opcional) y ejecutar el algoritmo principal. A diferencia de SAMODS y SAGAMODS, EMODS evita la convergencia lenta del método Simealing Annealing. EMODS explora diferentes regiones del espacio de soluciones factibles y busca soluciones no dominadas mediante búsqueda Tabú.

AERSMIDA: Desarrollado por Nieto,(2011) la estrategia que se plantea evita caer en óptimos locales al utilizar dos técnicas y a la vez permiten una diversificación de la población: Aplicando el operador genético de recombinación de la población (cruzamiento) y la técnica de recocido simulado, basándose en el algoritmo Simulated Annealing Metaheuristic of Deterministic Swapping (SAMODS). Cuando se obtienen las soluciones iniciales con o sin mejoras, se recorre el Autómata Finito Determinista Multi-Objetivo (AFDM) con el fin de buscar estados o soluciones no dominadas.

EMSA: Swapping, Theory of Evolution and Simulated Annealing metaheurística hibrida basada en la teoría de autómatas determinista Swapping, Theory of Evolution and Simulated Annealing para la optimización de problemas multi-objetivo combinatorios (Niño, Nieto, et al., 2011), es una mejora del algoritmo MODS, A diferencia de MODS, EMSA trabaja usando una dirección de búsqueda dada a través de la asignación de pesos para cada función a optimizar del problema combinatorio también, para evitar óptimos locales, EMSA usa como estrategia un algoritmo genético.

ERNEAD: ***E**ntrenamiento de **R**ed **N**euronal **E**volutiva **A**utómata **D**eterminista* implementada por Ruiz, (2013)entrenamiento de redes neuronales evolutivas mediante estrategias evolutivas y

autómatas finitos. Es una metaheurística basada en el algoritmo EMDOS, el cual fue utilizado para entrenar redes neuronales artificiales con el fin de llevar a cabo la evolución de los pesos sinápticos de las neuronas. La aplicación de esta metaheurística en las redes neuronales generará conjuntos de redes con pesos óptimos para un problema particular.

Como lo había mencionado, entre las metaheurísticas descritas se adoptará la meta-heurística ébola con autómata finito para el desarrollo de la investigación. De hecho, son pocas las investigaciones y publicaciones referentes a ella por lo tanto no ha sido aplicada fuertemente en la resolución de problemas reales.

2.5. Lógica difusa (Berlanga, 2010), (Hernandez, Ramirez, & Ferri, 2004).

La lógica difusa permite modelar conocimiento impreciso y cuantitativo, así como, como transmitir, manejar incertidumbre y soportar en una extensión razonable el razonamiento humano de una forma natural. Desde que Zadeh, (1965) propuso a teoría de conjunto difuso se ha aplicado en múltiples áreas de investigación, fundamentalmente por su cercanía y al razonamiento humano y por proporcionar una forma efectiva para capturar la naturaleza aproximada e inexacta del mundo real.

Como lo señaló en su trabajo Zadeh, (1973),las técnicas convencionales para el análisis de sistemas son intrínsecamente inadecuadas para tratar con sistemas humanistas, cuyo comportamiento está fuertemente influenciado por el juicio humano, la percepción y las emociones. Esta es una manifestación de lo que podría llamarse el "principio de incompatibilidad". A medida que aumenta la complejidad de un sistema, nuestra capacidad de hacer afirmaciones precisas pero significativas sobre su comportamiento disminuye hasta que se alcanza un umbral más allá del cual la precisión y la importancia se convierten en características casi mutuamente excluyentes" (L.A. Zadeh, 1973). Debido a esta creencia, Zadeh, (1971) propuso el concepto de variables lingüísticas como un enfoque alternativo para modelar el pensamiento humano.

En sí, un conjunto difuso es una entidad más compleja que un conjunto clásico, pero constituye una representación con mayor precisión para conceptos reales. Esta expresividad permite simplificar reglas y los sistemas basados en ellos.

Por su parte, una partición difusa de una variable determina una distinción de niveles en los valores de la misma mediante conjuntos difusos y permite solapamiento en las fronteras, al igual que ocurre en el razonamiento humano cuando trabajamos con valores lingüísticos (Zadeh, 1975). Así, una variable se describe mediante distintos términos lingüísticos de los que se utilizan habitualmente en el razonamiento humano con variables numéricas (por ejemplo, *Bajo, Medio, y Alto*). Cuando utilizamos estos conceptos, mentalmente pensamos en ellos con un cierto solapamiento entre algunos términos. De ahí que, cuando decidimos trabajar con una variable considerándola una variable lingüística que toma como valores términos lingüísticos, definimos una partición difusa que considere siempre una división del dominio en términos lingüísticos con cierto nivel de solapamiento. El significado de cada término viene especificado por un conjunto difuso y por tanto por una función de pertenencia. Esta función de pertenencia determinará de forma precisa para cualquier valor de la variable, el grado de pertenencia al conjunto difuso correspondiente.

Por otro lado, los sistemas basados en reglas que utilizan conjuntos difusos para describir los valores de sus variables se denominan sistemas basados en reglas difusas. Ante una situación dada, la regla se podrá aplicar si el antecedente de la misma describe la zona del espacio a la que pertenece el ejemplo, es decir, si el grado de compatibilidad del antecedente de la regla y el ejemplo es mayor que cero. Este grado se calcula de la siguiente forma:

Para cada variable se calcula el grado de pertenencia al conjunto difuso correspondiente. Si en la regla, para una variable se indica una disyunción de términos (por ejemplo, *Nivel = Bajo O Muy Bajo*), esto se interpreta como la unión de los conjuntos difusos correspondientes y el grado de

pertenencia se calcula con una t-conorma (operador de unión difusa) que habitualmente es el máximo, aunque existen otras definiciones para el mismo. Según esto el grado de pertenencia de un valor actual de la variable nivel a *"Bajo O Muy Bajo"* será igual al máximo entre el grado de pertenencia del valor actual del nivel a *Bajo* y el grado de pertenencia a *Muy Bajo*. Las reglas construidas de esta forma se denominan reglas en *forma normal disyuntiva* (*DNF, Disjunctive Normal Form*).

Habitualmente el antecedente de una regla difusa está formado por una conjunción de condiciones para las distintas variables. En este caso, el grado de compatibilidad del ejemplo con la regla se calcula con una t-norma (operador de intersección difusa). Existen múltiples expresiones posibles para la t-norma, pero es frecuente el uso del operador mínimo o producto.

Si utilizamos el mínimo, el grado de compatibilidad se calcula como el mínimo entre los grados de pertenencia de las variables implicadas en el antecedente a los conjuntos difusos correspondientes. Como se puede observar, si se elige la t-norma mínimo o producto, cuando una de las variables no pertenece al conjunto difuso implicado, el grado de compatibilidad del ejemplo con la regla es cero. Las reglas construidas de esta forma se denominan reglas canónicas.

Si en el antecedente las condiciones para las variables se combinan con el operador de disyunción (por ejemplo, *Nivel = Bajo O Edad = Joven*), se utilizará el operador de unión difusa (t-conorma) para el cálculo del grado de compatibilidad.

Cuando en el antecedente de las reglas sólo hay conjunciones de condiciones formadas por pares variables/valor, éstas se denominan reglas canónicas, por ejemplo.

SI (Zona=Norte Y Mejora imagen = Media Y Distintas alturas = No) ENTONCES Eficiencia = Baja

Sin embargo, si en estas condiciones aparecen disyunciones en los valores de cada variable, se denominan reglas normal disyuntiva (*DNF, Disjuntive Normal Form).* Ver ejemplo.

SI (Zona= (Norte O Sur) Y Mejora imagen = (Media O Alta) Y Distintas alturas = No) ENTONCES Eficiencia = Baja.

Dadas sus características las reglas difusas se pueden considerar como modelos locales simples, lingüísticamente interpretables y con un rango de aplicación muy amplio. Permiten la incorporación de toda la información disponible en el modelado de sistemas, tanto de la que proviene de expertos humanos que expresan su conocimiento sobre el sistema en lenguaje natural, como de la que tiene su origen en medidas empíricas y modelos matemáticos.

Es necesario destacar que no todos los sistemas difusos son sistemas basados en reglas difusas. Los conjuntos difusos se utilizan también, por ejemplo, en algoritmos de agrupamiento con el objetivo de obtener conjuntos difusos que permitan diferenciar los grupos con mayor expresividad, o también en algoritmos de modelado para predicción o regresión. Por ejemplo, se puede obtener un modelo que utilice conjuntos difusos para representar el conocimiento extraído, pero no necesariamente mediante reglas difusas.

En minería de datos, uno de los aspectos que determinan la calidad del conocimiento extraído, y por tanto del algoritmo utilizado, es la interpretabilidad del mismo y en este sentido los sistemas basados en reglas difusas son los más utilizados dentro del campo de la lógica difusa. De hecho, el uso de la lógica en minería de datos viene determinada por las siguientes razones:

- Cualquier proceso de minería de datos tiene como objetivo principal la identificación de patrones interesantes y la descripción de ellos de una forma concisa y con significado (Fayyad et al., 1996a).
- Los modelos difusos representan una descripción de los datos de entrada del usuario a través de un conjunto de reglas cualitativas que se establece en relaciones significativas y útiles entre variables.

- Los conjuntos difusos permiten establecer límites flexibles entre los distintos niveles de significado, sin ignorar ni enfatizar en exceso sólo los elementos cercanos a las fronteras, de igual forma que ocurre en la percepción humana.
- En todo proceso de extracción de conocimiento hay un componente de interacción humana y los conjuntos difusos permiten representar el conocimiento de forma lingüística, incorporar conocimiento previo de forma fácil y proporcionar soluciones interpretables.

En muchos algoritmos de minería de datos, y especialmente en la evaluación de los patrones extraídos, aparece el concepto de interés (Freitas, 1999), que agrupa distintas características como validez, novedad, utilidad, simplicidad y que puede cuantificarse a través de conjuntos difusos. De forma adicional, algunos algoritmos incorporan como parámetro adicional el interés, el cual es una medida de diferenciación difusa entre un patrón descubierto y un vocabulario definido por el usuario.

2.5.1. Sistema de clasificación basados en reglas difusas (Cordon. o. et al., 2004).

Un sistema de clasificación basado en reglas difusas (SCBRD) está compuesto por dos componentes principales:

1. Una Base de Conocimiento (BC) que contiene las reglas de clasificación difusa, y
2. un Método de Razonamiento difuso (MDR) que clasifica un nuevo patrón, es decir determina la clase asociada a él, usando información proporcionada por la BC, véase la figura 23.

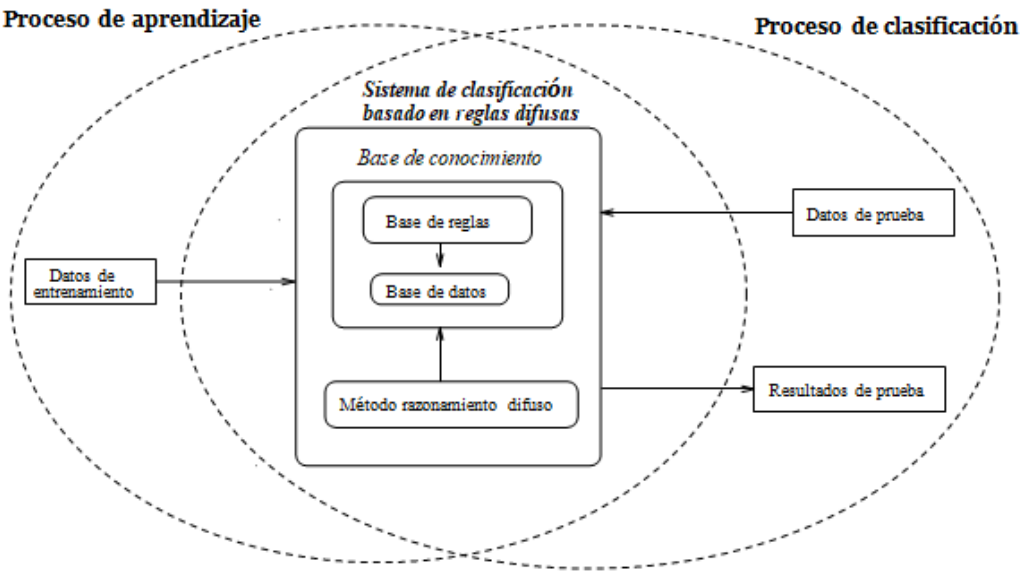


Figura 23. Diseño de un SCBRD
Nota: Adaptado de (Cordon. o. et al., 2004)

2.5.1.1. La base de conocimiento:

En un SCBRD la base de conocimiento (BC) está compuesta de la Base de Datos (BD), que contiene las funciones de pertinencia de las particiones difusas asociadas a las variables de entrada y la Base de Reglas (BR), constituida por el conjunto de reglas difusas.

- **La Base de Datos (BC)** cada variable *i* en el problema tendrá asociada una partición difusa de su dominio de definición, la cual representa los distintos conjuntos difusos asociados a cada uno de los distintos términos lingüísticos $L= L^m / m= 1$. La figura 24 muestra un ejemplo de una partición difusa de Ruspini con siete etiquetas lingüísticas y conjuntos difusos triangulares.

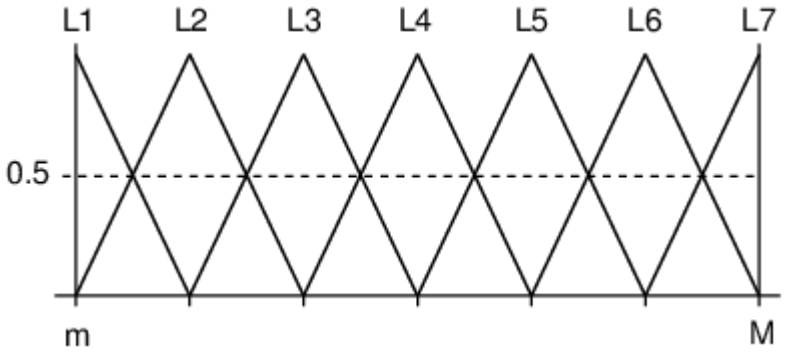


Figura 24. Ejemplo de partición difusa
Nota: Adaptado de (Cordon. o. et al., 2004)

- **La base de Reglas** (BR) formada por un conjunto de reglas de clasificación

$R = R^k / \ k = 1, \ldots, n_r$(siendo $n_r$el número total de reglas que forman la BR)

(a) Reglas difusas con una clase en el consecuente.

Si X_1es A_1 y ... y $X_n$es A_n **(59)**
entonces *Clase Y*es C_i

En donde $\boldsymbol{X_1}$……., $\boldsymbol{X_n}$ son variables asociadas a los diferentes atributos del sistema de clasificación, $C_i(i = 1, \ldots k)$son las etiquetas lingüísticas utilizadas para discretizar los dominios continuo de las variables.

($A\ A_k = L^m / \ m = 1$,...,l_i), y *Clase* es la variable que indica la clase C_ka la cual pertenece el patrón

(b) Reglas difusas con una clase y un grado de certeza en el consecuente.

Si X_1es A_1 y ... y $X_n$es A_n **(60)**
entonces *Clase Y*es C_i $con\ r$

Reglas difusas con un grado de certeza para todas las clases en el consiguiente

En donde e r es el grado de certeza de que un objeto que coincide con el antecedente de la regla pertenece a la clase C_i.

Este grado de certeza se puede determinar por la relación $\frac{S_i}{S}$ del número de objetos S_i en el subespacio definido por el antecedente de la regla que pertenece a la clase $C_i$al número total S de objetos en esa región.

(c) Reglas difusas con un grado de certeza para todas las clases en el consecuente.

$$\text{Si } X_1 \text{es } A_1 \text{ y } \dots \text{y } X_n \text{es } A_n \qquad \textbf{(61)}$$
$$\text{Entonces } (r_1, \dots, r_k)$$

En donde r_i $(i = 1, \ , k)$ es el grado de solidez para predicción de la clase $C_i$para un objeto en la región descrita por el antecedente. Los grados de certeza se pueden determinar utilizando la misma proporción considerada para las reglas de tipo b). Observe que la regla de clasificación de tipo c incluye los tipos a y b seleccionando los siguientes valores para $(r_1, \dots, r_k)$ el caso.

$$r_h = 1, \ r_i = 0, \ i \neq h, \ i = 1, \dots, k\text{=}1 \qquad \textbf{(62)}$$

Describe una regla de tipo (a), mientras que

$$r_h = r, \ r_i = 0, \ i \neq h, \qquad i = 1, \dots, k \qquad \textbf{(63)}$$

Es idéntica a una regla tipo (b).

Por otra parte, como el tamaño de la BC aumenta con el número de variables y términos lingüísticos, se requieren de términos lingüísticos adicionales para controlar la granularidad y por consiguiente la interpretabilidad del sistema. Existe un tipo de representación de reglas que permite obtener SCBRDs con alto grado de legibilidad al proporcionar una descripción más compacta de las relaciones difusas especialmente el antecedente de las reglas. A las reglas obtenidas mediante

este tipo de representación se les denomina reglas en forma disyuntiva (o reglas DNF), las cuales tienen la siguiente forma:Si X_1es A_1 y … y $X_n$es A_n entonces *Clase Y*es B,

Donde cada variable de entrada X_i toma como valor un conjunto de términos lingüísticos A, cuyos miembros están unidos por un operador disyuntivo, mientras que la variable de salida sigue siendo una variable lingüística habitual con una sola etiqueta asociada.

Por lo tanto, la sintaxis completa para el antecedente de la regla es.

$$\textbf{Si } \boldsymbol{X_1}\textbf{es } \boldsymbol{A_1} = \{\boldsymbol{A_{1l}}\ \boldsymbol{or} \ldots \boldsymbol{or}\ \boldsymbol{A_{1l_1}}\}\textbf{y} \ldots \textbf{y } \boldsymbol{X_n}\textbf{es}\boldsymbol{A_n} = \{\boldsymbol{A_{n1}}\boldsymbol{or} \ldots \boldsymbol{or}_{\boldsymbol{Anl_n}}\} \quad \textbf{(64)}$$

El uso de reglas DNF permite cambios en la granularidad del sistema al combinar términos lingüísticos mediante el operador o, dado, que permite la ausencia de alguna de las variables de entrada en cada regla (al conseguir que A_i^k sea igual al todo el conjunto de etiqueta lingüísticas de la variable i).

Cabe señalar que la representación en forma normal disyuntiva, puede combinarse con cualquiera de los tipos de reglas mencionados en el apartado anterior, siendo la escogida la presentada en la expresión (64) la conformada por el tipo c. No obstante, hay que advertir que en el estudio se utilizará las reglas DNF del tipo (b).

2.5.1.2. Método de razonamiento difuso.

En términos generales, el método estándar para construir un sistema de inferencia difuso depende de la potencia del razonamiento difuso para la obtención de un resultado en este caso una clasificación de reglas.

En este aspecto, diversas estudios han utilizado el método de razonamiento (MDR) del máximo, conocido también como MDR clásico o de la regla ganadora (Ishibuchi. H, Nozaki. K, & Tanaka. H, 1992), que considera la clase indicada por una sola regla aquella con la que

el ejemplo tiene mayor grado de compatibilidad (calculando por medio de la t-norma aplicado al grado de pertinencia del ejemplo a los distintos conjuntos difusos presentes en el antecedente). Por otra parte, también otros MDRs combinan la información aportada por todas las reglas que representan el conocimiento de la zona a la que pertenece el ejemplo (Cordón. O., del Jesus. M., & Herrera. F, 1999), (Cordón. O. et al., 1999).

A continuación, Berlanga (2010) describe el modelo general de razonamiento difuso. Los pasos de un MDR son los siguientes:

1. Calcular el grado de emparejamiento del ejemplo con el antecedente de las reglas.
2. Calcular el grado de asociación del ejemplo a la clase consecuente de cada regla mediante una función de agregación entre el grado de emparejamiento y el grado de certeza de la regla con la clase asociada.
3. Determinar el grado de asociación del ejemplo con las distintas clases
4. Clasificación para ello es necesario aplicar una función de decisión f sobre el grado de asociación del ejemplo con las clases, que determinara en base al criterio del máximo, a clase C a la que corresponde el mayor valor.

2.5.1.3. Tareas en el aprendizaje de los sistemas de clasificación basados en regla difusas.

Berlanga (2010) establece que el aprendizaje de un SCBRD consiste en suministrar un conjunto de ejemplos correctamente clasificados, y que al mismo tiempo determine con el mínimo error posible, la clase para un conjunto de ejemplos no clasificados.

El análisis de los componentes y del funcionamiento de un SCBRD permite determinar que el proceso de aprendizaje radica principalmente la realización de dos tareas complementarias entre sí:

- El establecimiento de las particiones difusas para los dominios de las

variables del problema, y

- La generación de un conjunto de reglas difusas.

Ambas tareas se engloban dentro del proceso de generación de la BC, el cual debe obtener un conjunto de reglas difusas que representen, de la forma más precisa posible las relaciones entre las variables del problema de clasificación y permitan la clasificación correcta de unos nuevos ejemplos que se representen al sistema.

La primera de las tareas consiste en la obtención del conjunto de etiquetas lingüísticas (y conjuntos difusos asociados) óptimo para cada variable.

- Estableciendo una partición difusa fija para cada variable en base a conocimiento experto o a una división equidistante del dominio con un tipo de función de pertenencia (triangular, trapezoidal, gaussiana) preestablecida o determinada tras una experimentación. Algunas propuestas clásicas donde es posible encontrar este enfoque son la extensión a problemas de clasificación del proceso de generación de Wang y Mendel propuesto por Chi y otros (Chi Z. Yan H. Pham T., 1996). O en el método de generación desarrollado por Ishibuchi y otros(Ishibuchi. H et al., 1992).
- Utilizando técnicas de agrupamiento que determinan grupos de datos con determinadas relaciones entre las características y construyendo los conjuntos difusos mediante proyecciones. Algunos métodos se pueden encontrar en (Castro. y Camargo, 2004), (Bouchachia.y Mittermeir, 2007), (Chi.y Yan,1996).

- Utilizando distintas técnicas de particiones difusas para cada variable, que se diferencian en el número de términos utilizados, como en Ishibuchi y otros, (Ishibuchi., Nozaki., & Tanaka., 1994), (Ishibuchi .H., Yamamoto. T., & Nakashima.T, 2005), (Ishibuchi .H. et al., 2005).

Con relación a la segunda de las tareas, la literatura especializada también muestra que es posible utilizar diferentes métodos para la generación de un conjunto de reglas difusas:

Métodos iterativos tales como:

- **Métodos basados en redes neuronales.** También conocidos como métodos o sistemas neuro-difusos (Nauck D. Kruse R., 1997), (Castellano G., Fanelli. A., & Mencar.c, 2002), (Chakraborty. D. & Pal N.R., 2004)

- **Métodos basados en técnicas de agrupamiento.** (Abe. & Lan., 1995), (Abe.S. & Thawonmas, 1997), (Castellano G. y Fanelli A. M., 2000), (Setnes M. Roubos H., 2000), (Wang J.-S. Lee C. S. G., 2000).

- **Métodos basados en la derivación de reglas difusas a partir de árboles de decisión.** (Chi. y Yan, 1996), (Mikut., Jäkel., & Gröll., 2005), (Abonyi., Roubos , & Szeifert., 2003).

- **Métodos basados en algoritmos evolutivos.** Las generaciones de un conjunto de reglas difusas se pueden tratar como un problema de optimización o búsqueda. Los AEs y, particularmente los AGs por su capacidad de búsqueda en espacios complejos han demostrado que son una herramienta de optimización robusta aplicable a esta y otras tareas que componen el aprendizaje de SCBRDs.

El aprendizaje de los distintos componentes de un SCBRD mediante un AEs ha dado lugar a los sistemas denominados Sistemas de Clasificación Basados en reglas Difusas Evolutivas (SCBRDs) los cuales se presentan a continuación.

2.5.2. Sistema de clasificación basada en reglas difusas evolutivos (Berlanga, 2010) , (Hernandez, Ramirez, & Ferri, 2004).

En esta última década, se han realizado muchas investigaciones en el área de la inteligencia artificial, y especialmente, en los sistemas difusos, el cual ha dado lugar a la integración de la lógica difusa con otras técnicas como los AEs (Bäck, 1996),las redes neuronales, razonamiento probabilístico. De hecho, un sistema difuso se puede abordar como un proceso de optimización o búsqueda y los AEs en particular los AGs, son considerados como la técnica de búsqueda global más implementada en la extracción de conocimiento. Debido a esta característica los AEs son notablemente indispensables para el diseño de los sistemas difusos, y su gran aplicabilidad dado lugar a la creación de los denominados sistemas difusos evolutivos.

Dentro de los sistemas difusos evolutivos, los más conocidos y estudiados son: Sistemas de clasificación basados en reglas difusas evolutivas (SCBRDs) (Casillas., Carse., & Bull., 2007), (F. Berlanga, 2010). Teniendo en cuenta el valor que aporta la integración de la lógica difusa con AEs, este estudio toma como punto de partida el diseño de un SBRDEs para el aprendizaje de la base de conocimiento.

Por otra parte, es necesario definir el esquema de codificación de la regla difusa dentro de la población de individuos. La literatura especializada ofrece dos enfoques a seguir:

1. **Enfoque "Cromosoma= Conjunto de reglas"**, también llamado enfoque Pittsburgh, en el que cada individuo representa un conjunto completo de reglas, de forma que los individuos compiten entre sí a lo largo del proceso evolutivo.

2. **Enfoque "Cromosoma= Regla",** En que cada individuo codifica una única regla y el conjunto completo de reglas se obtiene al combinar varios individuos de la población (esto es, cooperación entre reglas) o bien al combinar los individuos obtenidos en varias ejecuciones del proceso evolutivo (competición de reglas) de hecho, podemos encontrar tres propuestas genéricas con este enfoque:

    - **El enfoque Michigan:** En el que cada individuo codifica únicamente una regla pero la solución final será la población final o un sub conjunto de la misma, en este caso es necesario evaluar el comportamiento del conjunto de reglas al completo y la aportación de la regla individual al mismo (Casillas., et al., 2007).
    - **El enfoque IRL (Iterative Rule Learning):** En el que cada cromosoma representa una regla, y la solución global está formada por las mejores reglas obtenidas al ejecutar el algoritmo de una serie de iteraciones sucesivas (Cordón.O., del Jesus. M, F., & Lozano.M, 1999).
    - **El enfoque Cooperativo Competitivo (GCCL):** En el que toda la población o un subconjunto de ella codifica la base de reglas, en este modelo los cromosomas compiten y cooperan de forma simultánea, en este enfoque los individuos cooperan para la solución final y también compiten por su supervivencia (Pérez., Pérez, Rivera., Jesus., & López., 2010), (Rodríguez.C, Pena. M, & Piñero. P, 2015) adoptaron este enfoque de codificación.

No obstante, la elección del esquema de representación depende de la tarea a realizar por parte del algoritmo de minería de datos y, por tanto, del tipo de regla a descubrir.

Si el objetivo es determinar un conjunto de reglas de clasificación, entonces se debe evaluar el conjunto de reglas al completo, dicho esto, el enfoque que más se aproxima a lo deseado es

el esquema cromosoma= base de reglas, debido a la interacción que se establece entre las reglas. No obstante, este enfoque no es universal, y tiene dificultades cuando los individuos poseen una longitud superior, lo que conlleva a un incremento en el coste computacional del algoritmo y la modificación de los operadores genéticos, esto conlleva a explorar otros enfoque como el de Cromosoma= regla, que utiliza una sintaxis más reducida simplificando el diseño de los operadores genéticos, aunque este enfoque también tiene su grado de complejidad como el cálculo en el valor de la función de adaptación, ya que cada regla se evalúa individualmente, y es difícil determinar, la calidad del conjunto de regla al completo; por otra parte como el objetivo es obtener un conjunto de reglas, el algoritmo genético no deberá converger hacia un único individuo, para ello se incorpora una técnica de nichos para fomentar la existencia de individuos distinto dentro de la población.

Una vez presentados los esquemas de representación de reglas, a continuación, se detalla la codificación del antecedente y el consecuente en función del tipo de regla a obtener. De tal modo, que se puedan adaptar a los distintos tipos de enfoques de codificación mencionados.

Una regla se puede describir de forma genérica de la siguiente manera:

SI <condición1> y <condición2> y…y<condición N> ENTONCES <consecuente>

Dónde:

- Las condiciones expresadas en el antecedente pueden implicar atributos nominales o numéricos. Si es numérico se pueden emplear un proceso de discretización, o agrupamiento.
- El número de condiciones que aparece en el antecedente es variable.
- Cada una de las condiciones puede estar formada a su vez por una disyunción de

condiciones individuales.

- Si la regla es de clasificación, en el consecuente sólo aparecerá una condición que implica a un atributo, el atributo de clase o atributo objetivo, y éste no puede estar contenido en el antecedente.

- Una regla que determina dependencias funcionales se puede considerar como una generalización de una regla de clasificación, salvo que en el consecuente puede aparecer más de un atributo objetivo.

- Si la regla es de asociación no tiene porqué existir necesariamente una diferenciación entre atributos predictivos y objetivo, pudiendo aparecer más de una condición en el consecuente.

2.5.2.1. Operadores genéticos.

Además de los operadores genéticos clásicos que conocemos, existen distintas propuestas de operadores genéticos diseñados específicamente para AGs cuyo objetivo es el descubrimiento de reglas.

Para AGs con el esquema de representación "*Cromosoma = Regla"*, es necesario evitar la convergencia de la población a un individuo particular, ya que un solo individuo no es una solución válida para el problema. Mediante un operador de selección denominado *sufragio universal* que elige los individuos que deben participar en la recombinación mediante una elección por sufragio de los ejemplos de entrenamiento. Como se había mencionado se integra al proceso una estrategia de *nichos* que potencia la evolución de diferentes reglas que cubren diferentes zonas del espacio de búsqueda. De modo, que cada ejemplo vota por la regla que determina la zona del espacio de búsqueda en la que está

incluido, de forma proporcional a la medida de adaptación de la regla. Es una estrategia de *nichos* que potencia la evolución de diferentes reglas que cubren diferentes zonas del espacio de búsqueda.

Una de las operaciones más adecuadas en el proceso de descubrimiento de reglas es la operación de generalización-especialización que se lleva a cabo, básicamente, eliminando o añadiendo condiciones en el antecedente, que apoyan a los operadores de cruce, de mutación. Por otro lado, en el caso, de un AG en el que evolucionan reglas con distintos consecuentes no es conveniente intercambiar material genético entre individuos (reglas) con consecuentes distintos, pues se considera que es información que no está relacionada. Esto se puede solucionar con un operador de cruce que opere sólo dentro de sub-poblaciones, entendiendo que una subpoblación está formada por los individuos que representan reglas con igual consecuente.

2.5.2.2. Función de adaptación.

Con el fin de obtener reglas con capacidad predictiva alta, comprensibles e interesantes. El algoritmo evolutivo puede conseguir mediante una combinación lineal con pesos de estas tres medidas (Noda.; Freitas., & Lopes, 1999), como una matriz de confusión, orientar el proceso evolutivo hacia la búsqueda de un conjunto de soluciones no dominadas en los distintos objetivos marcados (Deb, 2001) Entendiéndose por soluciones no dominadas aquellas para las que no existe ninguna otra solución candidata que las supere en todos los objetivos marcados.

2.5.3. Sistemas evolutivos difusos en minería de datos (Berlanga, 2010), (Hernandez, Ramirez, & Ferri, 2004)

Siendo la lógica difusa una de las herramientas más utilizada en minería de datos por su carácter lingüístico de representar variables de forma expresable y fácilmente entendible, se puede extender y explotar su máxima representación combinándola con otras técnicas como las evolutivas para

obtener de ellas su mejor desempeño. A continuación, se definen los enfoques de la combinación lógica difusa algoritmos evolutivos.

- Algoritmo evolutivo que solo evolucionan reglas difusas, y utilizan funciones de pertenencia definidas por el usuario, es decir, permite incorporar al usuario en la especificación de las funciones de pertenencia.
- Algoritmos que solo evolucionan aspectos relacionados con las funciones de pertenencia.
- Algoritmos que evolucionan tanto reglas como funciones de pertenencia; esto se puede realizar de dos formas, una de ellas es utilizando individuos que incluyan los aspectos de las funciones de pertenencia junto con los valores de los atributos, otra forma con individuos distintos para los valores de los atributos y para los parámetros de la función de pertenencia, lo que implica la evolución en paralela de ambas poblaciones de individuos.

En (Cordón, Herrera, Hoffmann., and Magdalena, 2004) se puede encontrar una revisión en la literatura de los sistemas evolutivos difusos , en sus tres enfoques.

Esquema de representación:

En las particiones difusas predefinidas por el usuario se pueden utilizar los mismos esquemas de codificación que para reglas no difusas. Si estas reglas aprenden también las funciones de pertenencia, el esquema de codificación debe incluir su representación. En este caso, podemos optar por incluir las características de las funciones de pertenencia en la codificación de los individuos, o codificarlas de forma independiente en otro individuo que evolucione de forma separada.

En (Mendes., Voznika., Freitas, & Nievola, 2001) se detalla una forma de codificar las funciones de pertenencia utilizando un individuo separado dividido en K elementos, donde K es el número de atributos que se tratarán como variables lingüísticas y de los cuales se va a aprender las funciones

de pertenencia. Cada elemento consta de cuatro genes que definen colectivamente tres funciones de pertenencia (bajo, medio y alto) para el atributo correspondiente (ver Figura 25). Cada gen se utiliza para especificar las coordenadas de dos vértices del trapecio que pertenecen a un par de funciones de pertenencia adyacentes.

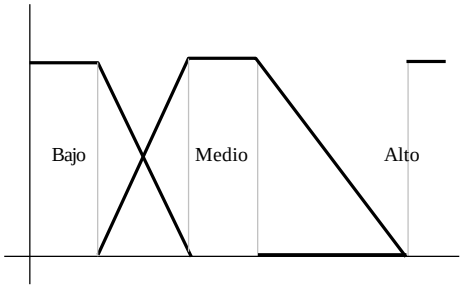


Figura 25. Definición de tres funciones trapezoidales de pertenencia mediante cuatro genes
Nota: Adaptado de (Berlanga 2010)

Esta representación tiene distintas ventajas: Reduce el espacio de búsqueda del AG, ahorra tiempo de procesamiento, forza a cierto solapamiento entre funciones de pertenencia adyacentes y garantiza que, para cada valor original del atributo continuo, la suma de sus grados de pertenencia a los tres valores lingüísticos sea 1.

Operadores genéticos.

Para la tarea de descubrimiento de reglas de asociación difusas, se pueden utilizar operadores genéticos clásicos (cruce, mutación, permutación.) adaptándolos a las características de la codificación utilizada para la representación de los individuos.

Función de adaptación (Hernandez, Ramirez, & Ferri, 2004)

.

A la hora de evaluar las reglas difusas obtenidas, se deben tener en cuenta aspectos como la precisión predictiva, la comprensibilidad y el interés.

Para medir **la precisión predictiva** de una regla difusa debemos calcular el grado de pertenencia de cada ejemplo al antecedente de la regla utilizando la t-norma y t-conormas elegidas para

representar los operadores de intersección y unión difusas. Una vez calculado el grado de pertenencia del ejemplo, la precisión predictiva del individuo (regla difusa) se puede calcular como (Romao, Freitas, and Pacheco, 2002a):

$$precisión = \frac{PredCorr \ - \frac{1}{2}}{TotPred} \quad \textbf{(65)}$$

Donde *PredCorr* (número de predicciones correctas) es la suma de los grados de pertenencia del antecedente de la regla para todos los ejemplos que tienen el valor predicho TotPred (número total de predicciones) es la suma de los grados de pertenencia al antecedente para todos los ejemplos.

Esta fórmula es una versión difusa de la medida de precisión utilizada por algunos algoritmos de minería de datos (Noda., Freitas, and Lopes, 1999). El objetivo al restar ½ a *CorrPred*en el numerador es penalizar las reglas que son demasiado específicas sin tener una influencia significativa en reglas más generales.

La precisión predictiva de una regla difusa también se puede medir calculando el factor de confianza, *FC*, a partir de las medidas *PV* (Positivos Verdaderos, suma del grado de pertenencia al antecedente de los ejemplos pertenecientes a la clase indicada en el consecuente), *PF* (Positivos Falsos, suma del grado de pertenencia al antecedente de los ejemplos pertenecientes a otra clase distinta a la indicada en el consecuente), *NF* (Negativos Falsos, suma del grado de pertenencia de los ejemplos a la zona del espacio no delimitada por el antecedente, pero que satisfacen el consecuente) y *NV* (Negativos Verdaderos, suma del grado de pertenencia a la zona no delimitada por el antecedente de los ejemplos no pertenecientes a la clase indicada en el consecuente). El factor de confianza se calcula con la siguiente expresión:

$$FC = \frac{PV}{PV + F} \quad \textbf{(66)}$$

También se puede extender la función de adaptación con una medida de comprensibilidad, que mida la simplicidad de la regla. Algunas funciones de adaptación que tienen en cuenta tanto la precisión predictiva como la comprensibilidad de una regla se describen en (Flockhart, and Radcliffe, 1995), (Giordana. & Filippo, 1996), (Janikow, 1993).

Otro criterio a considerar es **el interés.** En (Noda, Freitas., and Lopes, 1999) se propone una función de adaptación que tiene en cuenta de forma objetiva el grado de interés de una regla mediante una suma ponderada cuyos pesos son especificados por el usuario. Una forma de considerar las medidas de interés subjetivas en la función de adaptación se puede encontrar en (Romao, Freitas, and Pacheco, 2002b).

2.5.4. El problema de la alta dimensionalidad en el aprendizaje de Sistemas Basados en Reglas Difusas (Berlanga, 2010).

En el método de aprendizaje de SCBRDs, existen dos objetivos principales que deben intentar maximizar: la precisión y la interpretabilidad del conocimiento extraído en forma de reglas difusas. Muchas propuestas se enfocaron en mejorar la precisión de los SCBRDs, dejando a un lado la interpretabilidad. No obstante, algunos estudios recientes (Alcalá, Alcalá-Fdez., 2007), (Casillas, Cordón., Herrera., & Magdalena. 2003a), (Casillas., Cordón., Herrera., & Magdalena. ,2003b), (González., Rojas., Pomares., Herrera., Guillén., 2007), (Ishibuchi. , Nakashima. , & Nii. , 2004), (Ishibuchi. y Nojima., 2007), (Van Broekhoven, 2007), han puesto de manifiesto la necesidad de la existencia de un equilibrio entre la interpretabilidad y la precisión del conocimiento extraído a la hora de diseñar métodos de aprendizaje de SCBRDs.

Los autores mencionados sostienen que el equilibrio es más difícil de lograr cuando el problema que ha de resolverse presenta una alta dimensionalidad, es decir, un elevado nivel de variables o características de entrada y /o un elevado número de instancias no ejemplos.

Esta propuesta hace énfasis en aquellos problemas que representan alta dimensionalidad con respecto al número de variables de entrada, los principales inconvenientes en este tipo de problemas vienen dados por el crecimiento exponencial que se produce en el espacio de búsqueda de reglas difusas con un aumento lineal en el número de variables, conocido como explosión combinacional de reglas (Combs. W. E. & Andrews. J., 1998.), dicho crecimiento hace que el proceso de aprendizaje se vuelva más complicado, y el aprendizaje del SCBRD presente un elevado nivel de complejidad con respecto al número de reglas, variables y etiquetas incluidas en cada regla.

Para intentar solucionar esta desventaja se han propuesto varias alternativas con técnicas que están enfocadas en la reducción de ejemplos, el objetivo de estas técnicas es obtener un conjunto de ejemplos representativo del conjunto original con un tamaño inferior y con buena precisión de clasificación para los nuevos ejemplos de entrada. Estas técnicas la podemos encontrar en la literatura como técnicas de reducción, selección de ejemplos o características (García, Derrac, Cano, & Herrera, 2012).

En la literatura especializada se encuentran algunas alternativas de solución para que un SCBRDs presente un buen equilibrio entre interpretabilidad y precisión en problemas con alta dimensionalidad (Berlanga, 2010):

1. Llevar a cabo un proceso de selección de características:

Un determinado problema puede necesitar datos de distintas fuentes de información y si la cantidad de información de cada una de ellas es moderadamente grande la unión será inmensa.

En estos casos, es necesario aplicar un algoritmo de Selección de características (Bhatt., y Gopal, 2005), (Gunal y Edizkan, 2008), (Hu Q., Yu D., 2008), (Kohavi., y John.,1997), (Liu. y Yu, 2005), antes o durante el proceso de aprendizaje inductivo del SCBRD, que determine las características más relevantes para alcanzar el máximo rendimiento posible.

A continuación, se detallan algunos aspectos de la selección de características son:

- Menor cantidad de datos, de forma que se mejora la eficiencia del algoritmo de aprendizaje al poder aprender este más rápidamente.
- Mayor precisión, que hará que el SCBRD obtenido pueda generalizar mejor.
- Resultados más simples y más fáciles de entender, lo que mejorará la interpretabilidad del SCBRD obtenido.

Por otra parte, se distinguen dos tipos de algoritmos de selección de características, según la forma en que se evalúen los subconjuntos de características:

a) Filtro: Donde la evaluación de un subconjunto de características se hace en base a un criterio o medida que permite filtrar características irrelevantes, y que es independiente del algoritmo utilizado. Algunas de estas medidas son:
   - De información: Basadas en medidas de incertidumbre respecto a la clase verdadera.
   - De dependencia, asociación o correlación entre una característica dada y la variable de clase.
   - De distancia, separabilidad, divergencia o discriminación entre clases.
   - De consistencia, que tratan de encontrar el mínimo número de características que puedan separar las clases de la misma forma en que lo hace el conjunto completo de variables.

b) Envolvente: Donde la evaluación de un conjunto dado de características se hace en base a la precisión del clasificador introducido por un algoritmo de aprendizaje usando dicho subconjunto de características.

En la literatura especializada es posible encontrar procesos de selección de características basados en el uso de medidas de información (Bhatt. y Gopal, 2005), (Castellano.,& Castiello,2005), (Silipo y Berthold.,2000), (Shen. & Jensen , 2004), sistemas difusos evolutivos (Casillas Cordón, Herrera, & del Jesus, 2001), (González . Pérez, 2001), (Ho, & Chen, 2004), (Yu

S., 2002), sistemas neuro-difusos (Castellano et al., 2002), (Chakraborty. & Pal, 2004), (Li R., 2002), (Ravi., y Zimmermann, 2001), así como otras técnicas distintas (Ravi V., 2000), (Roubos., 2003).

2. Compactar o reducir un conjunto de reglas previamente aprendido, en una etapa de post-procesamiento:

Algunos procesos de aprendizaje de SCBRDs tienden a generar un número muy elevado de reglas difusas cuando el problema a resolver es de alta dimensionalidad, con la consecuente pérdida de interpretabilidad, para resolver este problema diversos investigadores han propuesto incluir un proceso de compactación y/o reducción del conjunto de reglas después del proceso de aprendizaje, el cuál minimizará el número de reglas incluidas en la Base de reglas y mejorará la interpretabilidad del SCBRD.

Los métodos de reducción de reglas actúan combinando o seleccionando un subconjunto de reglas del conjunto de reglas devuelto por el proceso de aprendizaje, con el objetivo de minimizar el número de reglas utilizadas mientras que se mantiene la precisión del SCBRD. Aquellas reglas que son conflictivas, que son innecesarias pueden ser eliminadas a través de este proceso de reducción de reglas, mejorando así tanto la interpretabilidad como el rendimiento de SCBRD.

En la literatura especializada es posible encontrar métodos de reducción basados en sistemas difusos evolutivos (Casillas., Cordón., del Jesus, & Herrera., 2005), (Ishibuchi., Nozaki., 1995), (Ishibuchi y Yamamoto., 2004),transformaciones ortogonales (Setnes M. y Babuška R., 2001), (Yen J. Wang L., 1999) y en medidas de similaridad (Abonyi , 2003), (Bouchachia A. Mittermeir R., 2007), (Chen. y Linkens, 2004), (Papadakis, y Theocharis, 2006), análisis de componentes principales ponderados y el análisis discriminante regularizados ponderados (Riaño. J., Prieto. F., Sánchez. E, C, & Castellano. G, 2010), (Al-Omairi, Abawajy, Chowdhury, & Al-Quraishi, 2019), (Aguilar. L, 2019), análisis discriminante lineal (Kalsoom, Maqsood, Ghazanfar, Aadil, & Rho, 2018), (Lu & Qiao, 2018), (Pham & Prakash, 2019), (Chen et al., 2019), (Mahdianpari et al., 2018),

análisis de correlación canónica (Alvarez, Boente, & Kudraszow, 2019), regresión lineal multivariada (Mori & Suzuki, 2018), (X. Zhang, Wang, & Wu, 2018).

3. Capítulo III

3.1. Un modelo para extracción de conocimiento en grandes bases de datos mediante cómputo evolutivo, autómatas deterministas y reglas difusas

En este capítulo se describe una propuesta para el aprendizaje de reglas, utilizando estrategias evolutivas en Sistemas de Clasificación Basados en Reglas Difusas (SCBRDs) para problemas con alta dimensionalidad, el algoritmo EKCEAD (A model for the extraction of knowledge in Databases through evolutionary computation, finite deterministic autómata and fuzzy rules). En problemas de alta dimensionalidad.

Se trata de un algoritmo evolutivo que aprende reglas en forma normal disyuntiva (DNF) generadas a través de una gramática independiente del contexto, y que utiliza el mecanismo de competición entre reglas para mantener la diversidad de la población, este mecanismo se encarga de la eliminación de las reglas "irrelevantes" durante el proceso evolutivo, lo que permite obtener reglas difusas que presenten una buena generalización.

Consecuentemente, se realizó un estudio experimental en problemas de clasificación con alta dimensionalidad, en ese estudio se comparan los resultados obtenidos por EKCEAD con los obtenidos por otros métodos de aprendizaje de SCBRDs, descritos en la literatura especializada.

Como parte final del estudio y en aras de suprimir la subjetividad inherente a la valoración de resultados. Se realiza una prueba de hipótesis utilizando el test estadístico de Friedman; se considera importante destacar que la precisión y confiabilidad del test se puede mejorar ejecutando los métodos en equipos de cómputo dedicados a la investigación.

3.2. Aspectos generales de la investigación

Este trabajo se centra en el desarrollo de un modelo basado en un algoritmo evolutivo para el aprendizaje de SCBRDs que equilibre precisión e interpretabilidad en problemas de alta dimensión, con respecto al número de variables de entrada. Generalmente la comprobación de dichas propiedades no es sencilla por lo que estos sistemas suelen presentar los siguientes inconvenientes:

- Son problemas del tipo NP-Hard (problemas de difícil resolución por métodos matemáticos convencionales)
- Experimentan el fenómeno conocido como explosión combinatoria, esto es una pequeña variación en el número de variables de entrada, se refleja en un gran cambio en el espacio de soluciones factibles.
- Al ser problemas de alta dimensionalidad, suelen presentar puntos de silla (máximos y mínimos locales) con lo cual se acrecienta el riesgo de convergencia prematura.
- Aprendizaje no generalizado, es decir, durante la fase de entrenamiento el algoritmo genera un conjunto de reglas, pero estas presentan poca aplicabilidad o confiabilidad al ser validadas con un nuevo conjunto de datos.
- Obtener un SCBRDs con alto elevado nivel de complejidad (lo que respecta al número de reglas, variables, etiquetas).

Como parte del proceso conceptual y de diseño del algoritmo evolutivo se plantearon las siguientes interrogantes.

1) ¿Cuál esquema es el adecuado para representar la solución?

   Existen dos enfoques con distintos tipos de propiedades para representar un conjunto de reglas difusas.

a) Enfoque “Cromosoma= Conjunto de reglas”, también llamado enfoque Pittsburgh, en el que cada individuo representa un conjunto completo de reglas, de forma que los individuos compiten entre sí a lo largo del proceso evolutivo
b) Enfoque “Cromosoma= Regla”, En que cada individuo codifica una única regla y el conjunto completo de reglas se obtiene al combinar varios individuos de la población.

No obstante, la elección del enfoque de representación dependerá de la tarea a realizar por parte del algoritmo de minería de datos y por tanto del tipo de regla a descubrir. En la tarea de clasificación, que es objeto de estudio de este proyecto. Si el objetivo es determinar un conjunto de reglas de clasificación, entonces se debe evaluar el conjunto de reglas al completo. En este sentido el enfoque que más se aproxima a lo deseado es el enfoque cromosomas = base de reglas (Pittsburgh) debido a la interacción que se establece entre las reglas, por esto, este enfoque ha sido el más frecuente en el diseño de SCBRDs. Sin embargo, este enfoque no es universal, y tiene dificultades cuando tiende a generar los individuos grandes y complejos, que decrementa la interpretabilidad de los SCBRDs obtenidos. Algunas propuestas para afrontar el problema han sido la introducción de operadores genéticos exclusivos que evitan la generación de individuos complejos, así como, determinar el tamaño máximo alcanzable por los individuos, mecanismo de poda para eliminar reglas irrelevantes. Sin embargo, todas estas soluciones conllevan a un incremento en el coste computacional del algoritmo, y un coste negativo sobre el nivel precisión debido a las restricciones impuestas a los individuos, o, lo que, es más, la complejidad del diseño.

Surge la necesidad de explorar otros enfoques como el enfoque cromosoma= Regla, el cual resulta más indicado para aquellas tareas de la minería de datos en las cuales el objetivo se centra en obtener un conjunto pequeño de reglas de alta calidad, en este caso las reglas son evaluadas

independientemente. Por ello utiliza una sintaxis más reducida simplificando el diseño con los operadores genéticos, aunque este enfoque también tiene su grado de complejidad como el cálculo en el valor la función de adaptación ya que cada regla se evalúa individualmente, y es difícil determinar la calidad del conjunto de reglas completo, por otra parte, como el objetivo es obtener un conjunto de reglas, el algoritmo genético no deberá converger hacia un único individuo.

Para abordar este problema, algunas propuestas recomiendan el enfoque IRL que consiste en obtener un conjunto de reglas, al ejecutar el algoritmo en una serie de iteraciones sucesivas. Sin embargo, como las reglas se obtienen en ejecuciones independientes hace que la competición entre ellas sea completamente nula, esto facilita que existan reglas redundantes o irrelevantes dentro del conjunto de reglas final e interferir en el nivel de precisión y de interpretabilidad de los SCBRDs (Cordón.O. et al., 1999).

Otra propuesta encontrada en la literatura es el enfoque GGCL, en este enfoque los individuos cooperan y compiten de forma simultánea para la solución final permitiendo de esta manera la eliminación de reglas redundantes, ciertamente aquí se obtiene un conjunto de reglas difusas con un buen nivel de precisión y de interpretabilidad (Pérez., Pérez, Rivera., Jesus., & López., 2010), (Rodríguez.C et al., 2015), por otra parte, es necesario la introducción un mecanismo (técnica de nichos) para mantener la diversidad en la población y evitar que los individuos converjan en la misma zona del espacio de búsqueda, también para fomentar la existencia de individuos distintos dentro de la población.

Otra posible solución, es el denominado enfoque Michigan en el que cada individuo codifica una única regla pero la solución final será la población final o un sub conjunto de la misma, siendo por tanto necesario evaluar el comportamiento del conjunto de reglas al completo y la aportación de la regla individual al mismo (Casillas., et al., 2007).

En concreto, en esta investigación utilizará como esquema de codificación de reglas el enfoque de Michigan.

2) ¿Cuál mecanismo es el adecuado para favorecer la diversidad dentro de la población?

En la literatura especializada se encuentran propuestas para favorecer la diversidad de individuos en una población. Estas propuestas generalmente se centran en el empleo de técnicas de nichos para evitar que los individuos converjan en una misma zona dentro del espacio de búsqueda. Entre las técnicas de creación de nichos se destacan los métodos de proporción o fitness sharing, los métodos de multitud o crowding y los métodos de aclarado o clearing (Pérez. E, Herrera. F., & Hernández. C., 2003) basados en medidas de similitud entre los individuos.

3) ¿Cuál medida de calidad se utilizará para calcular la bondad de la solución?

En la revisión de la literatura existente, se pudo verificar la existencia de variadas propuestas para medir la calidad de las reglas generadas, dichas mediciones pueden ser utilizadas para evaluar la bondad de una regla o de un conjunto de reglas. Se pueden clasificar entre aquellas que utilizan criterios subjetivos (requiere arbitraje humano) y objetivos (no requieren arbitraje). En particular, en el proceso de selección de reglas se utilizarán medidas de calidad objetivas.

Dado que en esta investigación se utilizó como esquema de codificación de reglas, el enfoque Michigan en el que cada individuo representa una única regla difusa, por lo tanto, se aplicará una medida de calidad fitness que permita evaluar la bondad de reglas difusas, dicha función de fitness debe incluir, medidas objetivas que evalúen:

- La precisión de una regla, esto es, la confianza de que el consecuente que muestra la regla es el correcto para aquellos ejemplos que emparejan con el antecedente de dicha regla.

- El porcentaje de acierto dentro del conjunto de ejemplos pertenecientes a la clase perteneciente en el consecuente de la regla.

4) ¿Qué tipo de operador genético que se debe utilizar?

   El principal operador genético empleado en cómputo evolutivo, específicamente en los algoritmos genéticos es el operador de cruce. Este operador intercambia material genético de dos padres con el objetivo de explorar el espacio de búsqueda y refinar los individuos obtenidos en cada momento, el cruce se aplica sobre los mejores individuos de la población, obteniendo dos nuevos individuos, que sustituirán a los dos peores individuos de la población. Por tanto, este operador es clave fundamental en el aprendizaje de reglas difusas, y es por eso que debe aplicarse con alta probabilidad.

   Por otro lado, el operador de mutación permite una exploración heurística del espacio. Generalmente la alteración aleatoria de uno de los componentes de código genético de un individuo provoca un salto en el espacio comenzando una búsqueda distinta en otra zona que puede ser más prometedora dentro del espacio de búsqueda, o menos prometedora. La mutación puede generar saltos evolutivos (acercarnos a la solución) o saltos involutivos (alejarnos de la solución), por ello, es conveniente emplear este operador, con una probabilidad reducida ya que ayuda a conseguir variedad genética, pero puede afectar negativamente la convergencia del método.

5) ¿Cuál esquema de reemplazamiento de la población se debe utilizar?

   Pueden presentarse situaciones en las cuales, se favorece la presencia de individuos poco favorables para la población, es decir que los buenos individuos desaparecieran a favor de otros peores, para solucionar este problema se debe elegir un esquema de reemplazamiento adecuado.

Existen varios esquemas de reemplazamiento de la población en el ámbito de los AG.

- El esquema estacionario, aquí se modifica solo una parte de la población, el procedimiento requiere el remplazar a uno de los padres por alguno de sus descendientes, esto quiere decir, que los peores individuos de la población son reemplazados por los nuevos descendientes con la condición de que estos últimos sean mejores.
- Esquema generacional, después de cada generación (padres) se reemplaza la población completa por sus descendientes (hijos), obteniendo de esta forma la siguiente generación.
- Esquema elitista, donde se utiliza una población intermedia con todos los padres y todos sus descendientes para seleccionar los mejores.

El esquema de reemplazamiento de la población utilizado en este estudio es el esquema elitista.

3.3. Modelo EKCEAD para la extracción de conocimiento en bases de datos a través del cómputo evolutivo, autómatas deterministas finitos y reglas difusas para problemas de alta dimensión

Se presenta el modelo evolutivo para el aprendizaje de SCBRDs en problemas con alta dimensionalidad denominado EKCEAD (A model for the extraction of knowledge in Databases through evolutionary computation, finite deterministic automata and fuzzy rules) ver algoritmo 3.

Sus principales características son las siguientes:

- Utiliza una gramática libre de contexto que aprende reglas DNF.
- Utiliza un autómata finito para guiar el proceso de ejecución del algoritmo.
- En este estudio para la obtención de reglas de clasificación utilizando PSO se ha decidido adoptar el enfoque Michigan, es decir que cada partícula de la población representa una

única regla, de tal forma que la base de regla está formada por todos los individuos de la población. Por lo tanto, es necesario definir función de aptitud (evaluación fitness).

ALGORITMO 3: Pseudocódigo de EKCEAD.

```
Input: P_M, P_C, NumIter, Estable, Azar, MaxIter
P_M ← 10
P_c ← 90
CargarCorrelaciones()
EscribirCorrelaciones()
GenerarPoblacionInicial()
NumIter ← 0
Estable ← 0
while NumIter < MaxIter ∧ Estable ≐ 0 do
    if Azar ≤ P_M then
        Mutacion()
    else
        Cruce()
    end
    NumInter ← NumIter + 1
    Estable ← CalcularEstabilidad
end
EscribirResultados
```

- EKCEAD incluye un mecanismo para mantener la diversidad de la población para evitar que todos los individuos converjan a una misma zona del espacio de búsqueda de reglas difusas. En particular se utilizó para ello la competición pseudo-elitista de modo, que las reglas compiten entre sí durante el proceso evolutivo, como consecuencia se eliminan las reglas irrelevantes para favorecer la generación de un número más reducido de reglas difusas que presentan una elevada pertenencia en clasificación.

- Para finalizar, EKCEAD utiliza un esquema de reproducción en que cada descendiente se genera aplicando los operadores genéticos de cruce y mutación. En donde padres como hijos compiten entre sí para la supervivencia y así formar parte de la nueva generación.

EKCEAD tiene varios métodos principales para su funcionamiento como se aprecia en la figura del algoritmo 3.

ALGORITMO 4: Pseudocódigo de Población Inicial.

```
1  for i = 1 to Nind do
2      j ← 0
3      while j < Ngenes do
4          Gen_sel ← Aleatorio() * NArt
5          Sw ← 0
6          for k = 1 to j do
7              if Gen_sel = Individuo(k) then
8                  Sw ← 1
9                  break
10             end
11         end
12         if Sw = 0 then
13             j ← j + 1
14             Individuo(j) ← Gen_sel
15         end
16     end
17     Poblacion(i) ← Individuo
18     Elite(i) ← Individuo
19 end
20 NElite ← Nind
```

Método Población Inicial: Este método genera la población inicial la cual se construye de forma aleatoria, al final el proceso devuelve la población inicial ver algoritmo 4.

Método cargar correlaciones: Este método se encarga de recuperar las ocurrencias relacionadas entre los artículos en la base de datos, Tiene como parámetros las siguientes variables:

- Artículos: Arreglo que guarda el código y el nombre del artículo
- Correlaciones: Arreglo bidimensional que almacena las ocurrencias relativas de un artículo con respecto al otro. Es una matriz simétrica con respecto a la diagonal principal por lo cual se optimiza el cálculo de la misma a solo la mitad.

ALGORITMO 5: Pseudocódigo de Cargar Correlaciones.

```
1  if not ConectarBD then
2  |  Escriba Error : No hay conexion
3  else
4  |  if not CargarArticulos() then
5  |  |  Escriba Error : No hay datos
6  |  else
7  |  |  Cart ← CantidadArticulos()
8  |  |  Correlaciones(Cart, Cart)
9  |  |  for i = 1 to Cart do
10 |  |  |  Cursor.Open(Compras Articulo por dia)
11 |  |  |  while not Cursor.Edf do
12 |  |  |  |  Cursor2.Open(ContarArticulos)
13 |  |  |  |  Correlaciones(I, BuscarArticulos) ← Cuenta
14 |  |  |  |  Cursor.Siguiente
15 |  |  |  end
16 |  |  end
17 |  end
18 end
```

El método en primer lugar recupera un listado con todos los artículos y los carga en un arreglo en forma de tuplas conformadas por el time_id y el custumer_id (compras de un cliente en un periodo) generando un cursor el cual realiza un proceso para relacionar las veces que cualquier otro artículo fue comprado el mismo día por el mismo cliente. Para

calcular la correlación se divide el número de coincidencias entre el número total de compras. Esta función es threadsafe por lo que se puede invocar múltiples veces para agilizar el proceso de los cálculos y utiliza la variable correlaciones como área de comunicaciones comunes. Ver algoritmo 5.

Función Aptitud (fitness): Este procedimiento se encarga de calcular la aptitud de un individuo, para lo cual se basa en la definición de individuo (cromosoma) y calcula las correlaciones específicas de cada artículo (gen) con todos los demás que conforman el individuo. La sumatoria de estas correlaciones será la aptitud del individuo $Fitness_i \sum_{j=1}^{n} Correlacion_{ij}$. Ver algoritmo 6.

ALGORITMO 6: Pseudocódigo de Función de Aptitud.

```
   Input: C_Gen
1  Suma ← 0
2  C_Gen ← individuo.length()
3  for i = 1 to C_Gen do
4  |   for j = 1 to C_Gen do
5  |   |   Fila ← BuscarFila(individuo(i))
6  |   |   Columna ← BuscarFila(individuo(j))
7  |   |   Suma ← Suma + Correlaciones(Fila, Columna)
8  |   end
9  end
10 Aptitud ← Suma
```

Tiene como parámetros las siguientes variables:

- Artículos: Arreglo que guarda el código y el nombre del artículo.
- Correlaciones: Arreglo bidimensional que almacena las ocurrencias relativas de un artículo con respecto al otro. Es una matriz simétrica con respecto a la diagonal principal por lo cual se optimiza el cálculo de la misma a solo la mitad.

- Individuo. Arreglo que guarda la definición del cromosoma, se constituye de n códigos de artículos.

Utiliza como función auxiliar Buscar fila que es una función a la cual se le suministra el código de un artículo y ubica la posición dentro del arreglo artículos, la cual servirá como índice para las correlaciones. Ver algoritmo 7.

ALGORITMO 7: Pseudocódigo Buscar Fila (Gen).

```
   Input: Gen
1  Cart ← Articulos.length()
2  for i = 1 to Cart do
3  |  if Articulo(i).Codigo = Gen then
4  |  |  Break
5  |  end
6  end
7  BuscarFila ← i
```

Método Seleccionar: Se implementa una versión de la selección por ruleta con la diferencia que incluye un 50% de la probabilidad que el individuo sea seleccionado sea del grupo elite, la forma de operar es la siguiente, ver algoritmo 8:

a) Se calcula la aptitud de la población (o grupo elite) mediante la sumatoria de aptitud de cada individuo.

b) Se calcula la probabilidad de que un individuo sea seleccionado como la proporción aptitud del individuo sobre aptitud de la población.

c) Se genera un arreglo que contiene los intervalos de selección para cada individuo.

d) Se genera un número aleatorio

e) Se ubica el intervalo al cual pertenece el número aleatorio.

Las variables que intervienen en el método son: Población, Elite, Slices. Ver algoritmo 8.

ALGORITMO 8: Pseudocódigo de Seleccionar.

```
1  SApt ← 0
2  Origen ← Aleatorio()
3  if Origen < 0,5 then
4      CCro ← Elite.length
5      for i = 1 to CCro do
6          Individuo ← Elite(i)
7          PApt ← Aptitud(individuo)
8          SApt ← SApt + PApt
9      end
10     LI ← 0
11     for i = 1 to CCo do
12         Individuo ← Elite(i)
13         PApt ← Aptitud(individuo)
14         Slice ← PApt/SApt
15         Slices(i).LI ← LI
16         LI ← LI + Slice
17         Slices(i).Ls ← LI
18     end
19     Seleccion ← Aleatorio()
20     for i = 1 to CCo do
21         if Seleccion ≥ Slices(i).LI ∧ Seleccion ≤ Slices(i).Ls then
22             Seleccionado ← Elite(i)
23             break
24         end
25     end
26 else
27     CCo ← Poblacion.length
28     for i = 1 to CCro do
29         Individuo ← Poblacion(i)
30         PApt ← Aptitud(individuo)
31         SApt ← SApt + PApt
32     end
33     LI ← 0
34     for i = 1 to CCo do
35         Individuo ← Poblacion(i)
36         PApt ← Aptitud(individuo)
37         Slice ← PApt/SApt
38         Slices(i).LI ← LI
39         LI ← LI + Slice
40         Slices(i).Ls ← LI
41     end
42     Seleccion ← Aleatorio()
43     for i = 1 to CCo do
44         if Slices(i).LI ≥ Seleccion ∧ Slices(i).Ls ≤ Seleccion then
45             Seleccionado ← Poblacion(i)
46             break                                    6
47         end
48     end
49 end
50 Seleccionar ← Seleccionado
```

Método Cruce: Se implementó un elitismo modificado a fin de garantizar la diversidad poblacional; además se incluyen conceptos propios de ébola para mantener un conjunto de individuos que sin ser parte de la población actual puedan ser utilizados en las diferentes operaciones genéticas (en esencia este grupo elite actúa como memoria poblacional para

poder ser utilizado en caso de desaparición de un individuo por alguno de los métodos aplicados). Ver algoritmo 9.

La modificación al cruce, representa una versión propia del proceso de investigación del cruce multipunto orientado por mascara, en la versión implementada de los n genes que conforman un individuo se verifica la existencia de elementos repetidos entre los padres, en caso de ocurrencia de esto, por ejemplo que existan k genes comunes (k<=n) se copian los k genes al hijo y los restantes (n-k) se toman aleatoriamente de un progenitor o del otro, es decir , una vez verificados los elementos que aparecen en ambos padres se pasan directamente al hijo, los restantes individuos se seleccionan por azar del padre1 o del padre2, es importante resaltar que el cruce no genera genes repetidos y una vez finalizado el cruce se aplica una estrategia de reducción, en el que se ubica el peor individuo y se sustituye por el hijo. En el caso del grupo elite sino se encuentra completo inserta directamente el hijo, y en el caso de que se encuentre completo el grupo elite se sustituye el peor individuo del grupo elite. Así mismo, la mutación para mantener la diversidad.

ALGORITMO 9: Pseudocódigo de Cruce.

```
Individuo1() ← Seleccionar()
Individuo2() ← Individuo1()
while Individuo1() = Individuo2() do
    Individuo2() ← Seleccionar()
end
NGenes ← Individuo1.length()
for i = 1 to NGenes do
    Buscar ← Individuo1(i)
    Sw ← 0
    for j = 1 to NGenes do
        if Buscar = Individuo2(i) then
            Sw ← 1
            break
        end
    end
    if Sw = 1 then
        HGen ← HGen + 1
        Hijo_HGen ← Buscar
    end
    for i = HGen + 1 to NGenes do
        SPadre ← aleatorio()
        if SPadre < 0,5 then
            for j = 1 to NGenes do
                Pasar ← Individuo1(j)
                for k = 1 to i do
                    if Pasar = Hijo(k) then
                        Sw ← 1
                        break
                    end
                end
                if Sw = 0 then
                    Hijo(i) ← Pasar
                end
            end
        else
            for j = 1 to NGenes do
                Pasar ← Individuo2(j)
                for k = 1 to i do
                    if Pasar = Hijo(k) then
                        Sw ← 1
                        Break
                    end
                end
                if Sw = 0 then
                    Hijo(i) ← Pasar
                end                                  4
            end
        end
    end
end
Peor ← 200
for i = 1 to Nind do
    if Aptitud(Poblacion(i)) < Peor then
        Peor ← i
    end
end
```

ALGORITMO 9: Pseudocódigo de Cruce.

```
                                /*  ALGOTIRMO 9:  CRUCE (CONTINUACION)  */
57 Poblacion(Peor) ← Hijo
58 Peor ← 200
59 if NElite < 1000 then
60 |   NElite ← NElite + 1
61 |   Elite(NElite) ← Hijo
62 else
63 |   for i = 1 to NElite do
64 |   |   if Aptitud(Elite(i)) < Peor then
65 |   |   |   Peor ← i
66 |   |   end
67 |   end
68 |   Elite(Peor) ← Hijo
69 end
```

ALGORITMO 10: Pseudocódigo de Mutación.

```
 1 Ind ← Seleccionar() * Nind
 2 Gen ← Seleccionar() * NGenes
 3 Sw ← 1
 4 while Sw = 1 do
 5 |   NueGen ← Aleatorio * NGenes
 6 |   NueGen ← Articulos(NueGen).Codigo
 7 |   Sw ← 0
 8 |   for i = 1 to NGenes do
 9 |   |   if Poblacion(Ind, I) = NueGen then
10 |   |   |   Sw ← 1
11 |   |   |   break
12 |   |   end
13 |   end
14 end
15 if Sw = 0 then
16 |   Poblacion(Ind, Gen) ← NueGen
17 end
```

Función Calcular Estabilidad: Esta función valora el momento cuando las poblaciones se han estabilizado para la cual usa el ECM (error cuadrático medio) de las ultimas 30 generaciones. En caso que el ECM se haga menor que el valor definido como tolerancia del método activa el criterio de fin. Utiliza como corrutina Aptitud con el fin de determinar la aptitud de la población. La forma como se comporta el método es como una cola con corrimiento hacia arriba, es decir cuando llega una nueva generación toda se desplaza una posición hacia arriba y el nuevo valor se inserta al final de la cola. Ver algoritmo 11.

ALGORITMO 11: Pseudocódigo de Calcular Estabilidad.

```
1  ECM ← 0
2  for i = 2 to 30 do
3  |  Homeo(i − 1) ← Homeo(i)
4  end
5  Suma ← 0
6  for i = 1 to NInd do
7  |  Suma ← Suma + Aptitud(Individuo(i))
8  end
9  Homeo(30) ← Suma
10 Suma ← 0
11 for i = 1 to 30 do
12 |  Suma ← Suma + Homeo(i)
13 end
14 Suma ← Suma ÷ 30
15 for i = 1 to 30 do
16 |  ECM ← (Suma − Homeo(i)) ∧ 2
17 end
18 CalcularEsatabilidad ← NumIter > 30 ∧ ECM ≤ Tol
```

Función de Reducción: Al final la reducción implementa una técnica elitista del tipo λ_{n+k}, esto es, luego de realizar cada operación genética se produce una población temporal que contiene los n progenitores y los k hijos y se selecciona de esta población lo n individuos más aptos (con mayor aptitud); en caso de existir dos o más individuos con igual aptitud y ante la carencia de operadores de protección de genes, se selecciona de forma aleatoria cualquiera de los cromosomas para pasar a la población.

Autómata: Proporciona la estabilidad, cuando la estabilidad se alcanza por el número máximo de iteraciones, como podría ser que el algoritmo se quede estancado en un máximo local, se puede utilizar la población de ese momento como población inicial para un proceso de refinamiento.

En esencia un autómata representa una máquina de estados finitos compuesta por una serie de estados (representado por los círculos) y un conjunto de acciones que deben pasar de un estado a otro.

El algoritmo EKCEAD inicia con la generación de la población inicial la cual se construye de forma aleatoria controlando que cada cromosoma no admita genes repetidos. Luego una operación genética hace la transición del estado inicial al estado de iteración llamado P_i, se establecen dos criterios de fin.

El primero de ellos estabilidad evalúa el Error Cuadrático Medio (ECM) de las aptitudes de las últimas 30 generaciones y si resulta ser menor que la tolerancia detiene el proceso. El segundo de ellos se basa en un número máximo de iteraciones, si el método alcanza este número sin haberse estabilizado también se detiene el proceso (para evitar que se quede en un ciclo infinito) ver figura 26.

El error cuadrático medio (ECM) mide la cantidad de error que hay entre dos conjuntos de datos. Es decir, compara un valor predicho y un valor observado o conocido(Ruiz, 2013).

.

$$\text{ECM} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \quad \textbf{(67)}$$

En donde,

$\hat{Y}_i$= Valores obtenidos de la predicción.

Y_i= Valores reales (medidos).

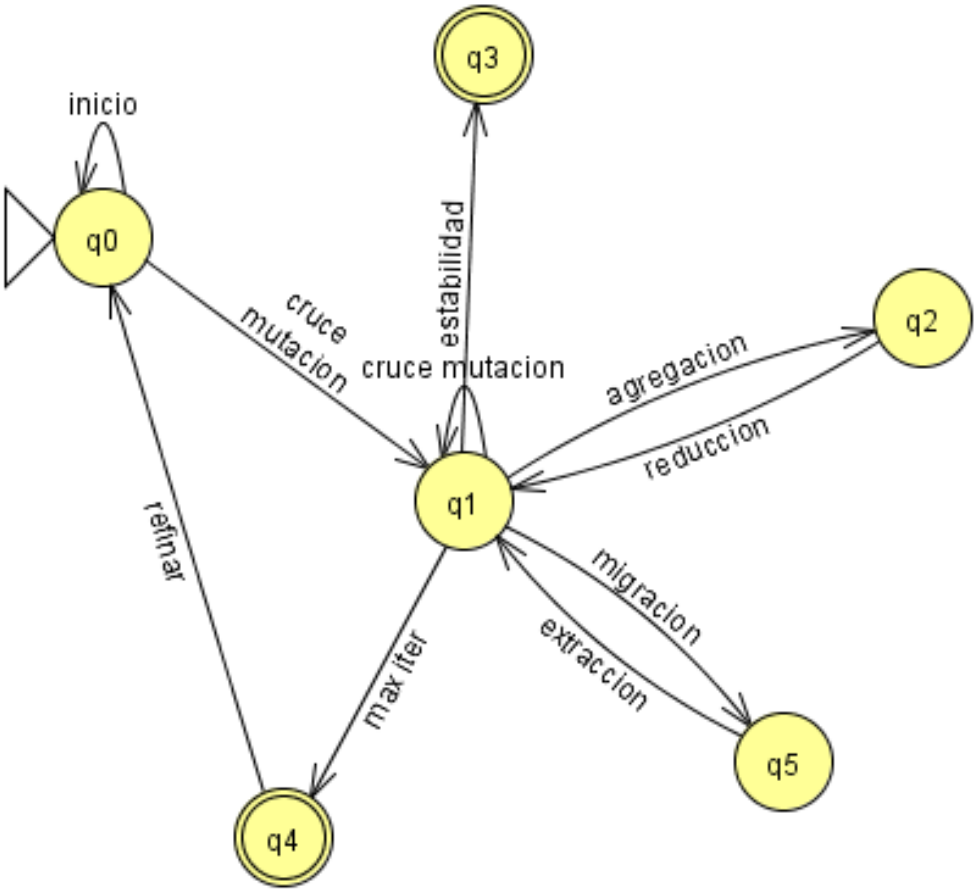


Figura 26. Entrenamiento autómata finito determinista
Fuente: Propia

Tabla 8. Matriz de Transición EKCEAD

Estado	refinar	cruce	mutación	Max iter	Migración	extracción	Reducción	Estabilidad	Agregación
q_o		q_1	q_1						
q_3									
q_4	q_o								
q_5						q_1			
q_2							q_1		
q_1		q_1	q_1	q_4	q_5			q_3	q_2

3.3.1. Definición de la gramática libre de contexto.

Como se ha indicado, EKCEAD utiliza una gramática libre de contexto para aprender reglas difusas en forma normal disyuntiva (DNF). Denotada en la siguiente expresión (Cordón, Herrera, Hoffmann, and Magdalena, 2001).

$$\textbf{Si } X_1 \textbf{ es } A_1 \textbf{ y } \dots \textbf{ y } X_n \textbf{ es } A_n \quad \textbf{Entonces clase Y es } C_i^r(r_1,\dots,r_k) \tag{68}$$

En donde cada variable de entrada X_i toma como valor un conjunto de etiquetas lingüísticas A_i unidas por el operador de disyunción (o), y donde r_i ($i = 1, \ , k$) es el grado de solidez para el predicción de la clase C_i para un objeto en la región descrita por el antecedente.

Por lo tanto, la sintaxis completa para el antecedente de la regla es.

$$\begin{aligned} \textbf{Si} X_1 \textbf{es} A_1 &= \{A_{1l} \textbf{or} \dots \textbf{or} A_{1l_1}\} \textbf{y} \dots \textbf{y} X_n \textbf{es} A_n \\ &= \{A_{n1} \textbf{or} \dots \textbf{or}_{Anl_n}\} \end{aligned} \tag{69}$$

Los grados de certeza se pueden determinar por la relación $\frac{S_i}{S}$ del número de objetos S_i en el subespacio definido por el antecedente de la regla que pertenece a la clase $C_i$al número total S de objetos en esa región.

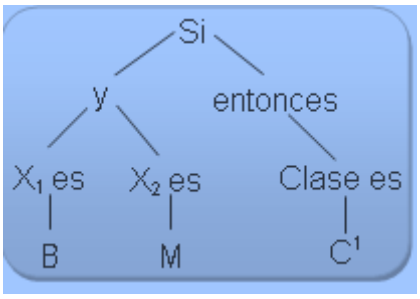


Figura 27. Codificación de una regla

Grado de certeza se obtiene como:

$$Confianza\ (A \rightarrow B) = \frac{\sum x_p \epsilon\ N\mu AB(x_p)}{\sum x_p \epsilon\ N\mu A(x_p)} \quad \textbf{(70)}$$

En donde |N| es el número total de ejemplos, $\mu_A(x_p)$ es el grado de emparejamiento del ejemplo x_p con el antecedente de la regla y $\mu_{AB}(x_p)$ es el grado de emparejamiento del ejemplo x_p con la regla completa.

3.3.2. Evaluación de la calidad de la solución.

Dentro de este marco, a cada uno de los cromosomas (reglas) que se encuentran en la población final (solución) se le aplican dos métricas Confianza y Cobertura- que permiten verificar de forma objetiva (sin intervención de un operador humano) la calidad de las reglas generadas. Dichas medidas son:

- La confianza denominada también precisión mide el porcentaje de veces que la regla se cumple

$$Confianza\ (A \rightarrow B) = \frac{\sum x_p \epsilon\ N\mu AB(x_p)}{\sum x_p \epsilon\ N\mu A(x_p)} \quad \textbf{(71)}$$

- Soporte denominado también Cobertura mide el número de instancias en las que la regla se puede aplicar y predecir correctamente.

$$Soporte\ (A \rightarrow B) = \frac{\sum x_p \epsilon\ N \mu AB(x_p)}{|N|} \quad \textbf{(72)}$$

En donde |N| es el número total de ejemplos, $\mu_A(x_p)$ es el grado de emparejamiento del ejemplo x_p con el antecedente de la regla y $\mu_{AB}(x_p)$ es el grado de emparejamiento del ejemplo x_p con la regla completa.

Para la presente investigación se utilizaron los datos de la compañía Foodmart,(una cadena de supermercados), se tomaron los datos del año 1997 para generar el conjunto de reglas difusas y los correspondientes a 1998 para evaluar el grado de cobertura y precisión de las reglas generadas, obteniéndose como resultados un grado de cobertura del 50% y un nivel de precisión del 45%, se hace necesario verificar con múltiples ejecuciones a fin de comprobar si no hay una relación significativa entre los datos de los períodos consecutivos o si fueron afectados por un agente externo (fenómeno de estacionalidad). Aunque los porcentajes de precisión son influenciados por la aleatoriedad, el resultado obtenido por el método propuesto predice un alto porcentaje de comportamiento de compra de los clientes.

3.3.3. Manteniendo la diversidad en la población: Competición elitista.

La implementación de cada operación genética ha resultado ser ligeramente modificada en la presente investigación, con el fin de mantener la diversidad, de forma tal que en la estrategia de selección (selección por ruleta), la aptitud de cada individuo está directamente relacionada con la probabilidad de ser seleccionado (a mayor aptitud mayor probabilidad de ser seleccionado), esto garantiza que aun el individuo con aptitud más baja (peor) tiene probabilidad de ser seleccionado, lo cual redunda en una forma de protección o permanencia de cromosomas.

3.3.4. Implementación de los operadores genéticos.

Según se ha discutido con anterioridad, cada uno de los operadores genéticos (principales) y secundarios resultó ligeramente modificado para satisfacer los requerimientos del problema, a continuación, se describe la forma como cada uno de ellos se implementó:

a. Selección. Elitista, basada en selección por ruleta con proporcionalidad en función de la aptitud de cada individuo con relación a la aptitud de la población.

$$Proporcion_i \frac{\sum Correlacion_{ij}}{\sum Cromosoma_i} \quad \textbf{(73)}$$

Es importante notar que antes de aplicar el proceso de selección se determina de donde se tomará el individuo, dicha determinación se realiza de forma aleatoria con probabilidad de 50% y 50% que provenga de la población actual o del grupo élite.

b. Cruce. Los individuos se evalúan en función de una máscara, pero antes se verifica la existencia de genes comunes entre ellos, en caso de encontrarse genes comunes, el primer paso es copiarlos directamente a la descendencia, para el cálculo de los restantes elementos, estos se seleccionan de forman aleatoria entre el progenitor 1 y el progenitor 2.

c. Mutación. La alteración se realiza de forma aleatoria en uno o más genes, el operador de mutación garantiza (a) cambio del gen y (b) no repetición del gen.

d. Reducción. La reducción utiliza un enfoque elitista, donde al finalizar cada iteración se agrupan los individuos existentes (progenitores) con los nuevos (hijos), y de esta agrupación se selecciona el número necesario de individuos (solo los mejores); para dirimir la posibilidad de empate entre individuos diferentes con igual aptitud se utiliza la aleatoriedad pura.

e. Migración. Se utiliza el fenómeno de la migración para construir un grupo de elite (EBOLA) el cual contiene 1000 individuos que durante las iteraciones del método han

demostrado tener una excelente aptitud; cuando uno de estos individuos aparece se crea una copia de él y es enviada al grupo élite; al interior del grupo élite la sustitución de individuos es directa; esto es, el nuevo individuo se ubica en la posición correspondiente a su aptitud y desplaza hacia abajo a los que le siguen, de forma tal que el último de ellos resulta eliminado.

3.3.5. Etapa de reproducción: Selección, aplicación de los operadores genéticos y reemplazamiento de la población.

Una de las consideraciones a tener en cuenta en todo proceso evolutivo consiste en indicar como se genera una nueva población de individuos a partir de la población actual de padres. Esto lleva consigo explicar la forma en que se seleccionan los padres, la forma en que se utilizan los distintos operadores genéticos, el número de hijos que se generan a partir de los padres y como se forma la nueva población usando los hijos.

Por otra parte, en cada iteración en la etapa de reproducción se genera un número de descendientes igual al tamaño de la población actual.

- Preparación: Se utiliza las ventajas de la inteligencia de enjambres para crear múltiples agentes de búsqueda encargados del cálculo de las correlaciones relativas entre los diferentes productos almacenados en la base de datos, es de aclarar que si bien a mayor cantidad de agentes se reduce el tiempo necesario para explorar la totalidad del espacio de búsqueda (todos los registros a nivel de la base de datos), generar una gran cantidad de ellos impone un sobrecosto adicional en lo referente a recursos computacionales; la presente investigación no se centra en determinar la relación ideal entre número de agentes y consumo de recursos computacionales, sobre todo si se tiene en cuenta que dicha relación resulta severamente afectada por la distribución de los datos (espacio de

búsqueda de soluciones), características del computador utilizado y cantidad de procesos que se encuentran ejecutando de forma simultánea en el equipo. El análisis de los datos almacenados reveló que no se individualizan de forma específica las compras realizadas, por lo que fue necesario suponer que cada pareja cliente/día representa una compra, suposición que si bien carece de validez completa (un cliente puede realizar varias compras el mismo día) no limita la generalidad de la solución (A. Gokil & S. Rajalakshmi, 2014), (Gupta, 2012), (Asadi, Afzali, Shojaei, & Sulaimani, 2012).

- Población Inicial (Po), la población inicial se genera de forma completamente aleatoria al crear n cromosomas cada uno de ellos con m genes, la generación de los cromosomas impide la presencia de genes repetidos, no así la creación de la población que puede producir cromosomas repetidos.
- Grupo élite inicial (EBOLA), inicialmente todos y cada uno de los individuos de la población son copiados en el grupo elite, es decir que en la iteración 0 tanto la población como el grupo élite son exactamente iguales; esta situación comienza a variar rápidamente de forma que en cada iteración se introducen nuevos cromosomas hasta completar la cantidad de 1000.
- Operadores genéticos: Se ha discutido con antelación la forma en que se implementaron cada uno de los operadores genéticos principales y secundarios y como cada una de las modificaciones planteadas se justifica en la necesidad de los algoritmos de minería de datos basados en reglas difusas de mantener la diversidad poblacional con el fin de evitar problemas de convergencia al estancarse en la presencia de mínimos locales; sobre este respecto se debe

destacar la presencia de la mutación con la implementación basada en un enfoque de probabilidad adaptativo, esto es luego de cada iteración, si no se producen mutaciones se reajustan las probabilidades de las operaciones incrementando en un 5% la probabilidad de ocurrencia de una mutación (para forzar la ocurrencia de una mutación cada 20 generaciones); tan pronto ocurra la mutación las probabilidades se restablecen a sus valores iniciales.

- Estrategia de reemplazo: Al finalizar cada operación genética la cantidad de individuos disponibles aumenta en el número de nuevos individuos generados; como restricción autoimpuesta del método la cantidad de individuos en a) la población y b) el grupo élite debe permanecer constante por lo cual se hace necesario reducir (eliminar una cantidad de individuos igual a los nuevos que se han generado). En el caso de la población se utilizó una estrategia elitista seleccionando los mejores individuos de la población, en el caso del grupo elite se utilizó una estrategia elitista con reemplazo, esto es si surge un nuevo individuo con mejor aptitud que alguno de los presentes en el grupo élite, se reordenan en función de sus aptitudes y se seleccionan los m mejores individuos.

3.3.6. Descripción del algoritmo.

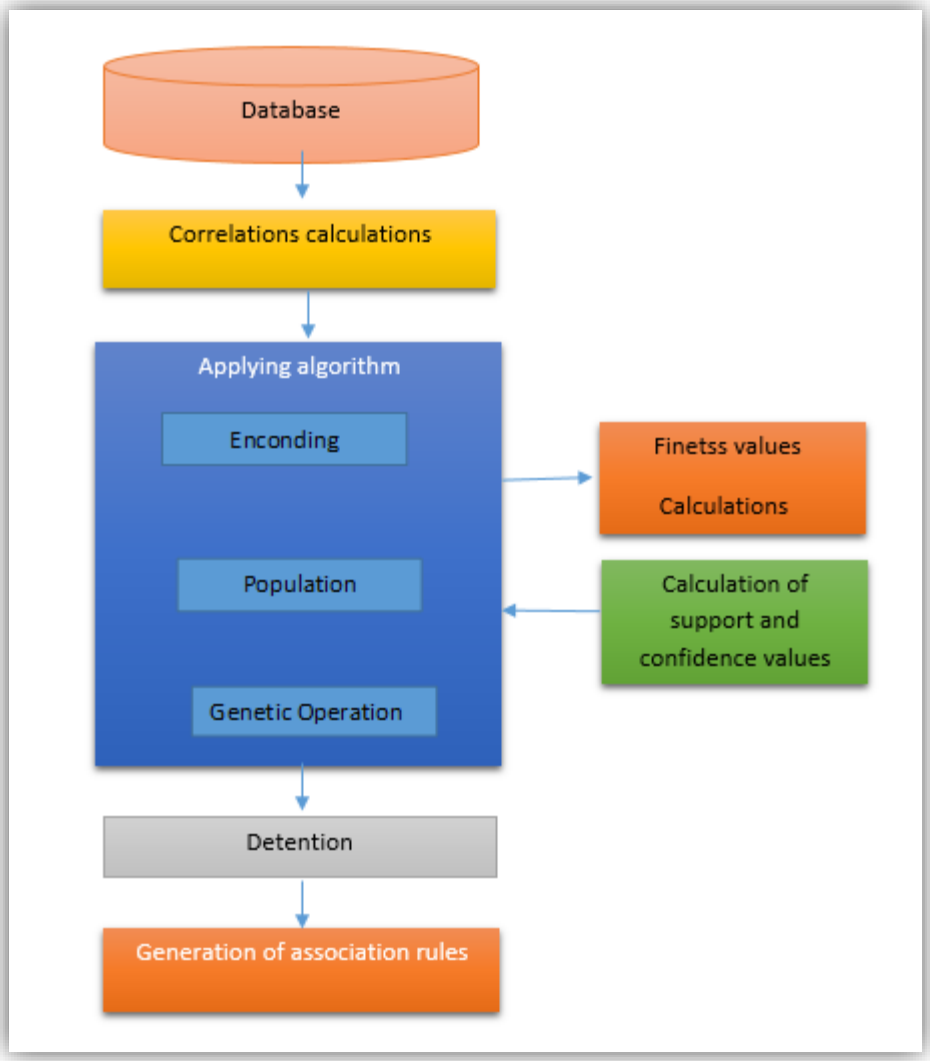


Figura 28. Marco general mediante EKCEAD
Fuente: Propia

Flujo del software

1. Se calcula la cantidad de artículos registrados en la base de datos.
2. Por cada artículo se determina la cantidad de comprar en las que aparece (dupla cliente – día).
3. Para cada una de las duplas encontradas se determina el número de veces que uno cualquiera de los otros artículos aparece (Correlación relativa).

4. Se carga la matriz de correlaciones relativas en un arreglo, dicho arreglo será la base para el cálculo de la función de aptitud.

5. Se genera la población inicial de tamaño 100**,** grupos aleatorios de 20 productos cada uno (cada grupo de productos es un cromosoma y cada producto es un gen).

6. Se evalúan todos los individuos de la población (el método de evaluación es el promedio de las correlaciones entre los 20 alelos y el total de las ventas)

$$\sum_{i,j=1,1}^{n,N} Prom.Correlacion_{Pi,Dj}\ Donde\ P \in Products\ escogidos \vee D \in Total\ productos \quad \textbf{(74)}$$

7. Se verifican cuales pueden entrar en el grupo elite (el grupo elite está conformado por los mejores 1000 individuos) y los que si se ingresan.
8. Se determina la siguiente operación genética a aplicar con base en la aleatoriedad y las probabilidades de ocurrencia de cada operación.
9. Se aplica la operación genética correspondiente.
10. Si resulta ser un cruce, entonces.
    a. Se seleccionan los progenitores en función de la aptitud. La escogencia de los padres depende del fitness [F], calculado mediante la sumatoria de las ocurrencias de los alelos (productos) de cada individuo. Aun cuando se escogen aleatoriamente tienen más probabilidad de ser escogidos aquellos individuos que tengan mayor fitness

b. Para el proceso de cruzamiento hay que tener en cuenta que el orden de los alelos no influye en la respuesta, pues representan productos escogidos, dentro de un grupo de productos totales.

c. Lo primero que se hereda son los alelos en común, en congruencia al punto b. o sea los productos que estén en ambos padres como indica la figura.

Padre						
Alelo	1	2	3	4 ...	19	20
Descripción del Alelo	Producto 80	Producto 952	Producto 224	Producto 530	Producto 332	Producto 128

Hijo						
Alelo	1	2	3	4 ...	19	20
Descripción del Alelo	Producto 952					

Madre						
Alelo	1	2	3	4 ...	19	20
Descripción del Alelo	Producto 776	Producto 12	Producto 591	Producto 138	Producto 952	Producto 158

Figura 29. Cruce de alelos

d. Los alelos ya usados de cada padre se marcan para no ser usados nuevamente.

e. Una vez se verifica que no existan más alelos (productos) en común en las soluciones, se procede a generar un aleatorio entre 0 y 1 para decidir que padre aportara el siguiente alelo.

f. Una vez decide que padre aportará el siguiente alelo se genera un aleatorio entre 1 y 20 cuantas veces sea necesario hasta conseguir un alelo, del padre escogido, que no haya sido usado antes y se le asigna al hijo. Ejemplo si el aleatorio escogió al padre y de este el alelo 4.

Padre						
Alelo	1	2	3	4	19	20
Descripción del Alelo	Producto 80	Producto 952	Producto 224	Producto 530	Producto 332	Producto 128

Hijo						
Alelo	1	2	3	4	19	20
Descripción del Alelo	Producto 952	Producto 530				

Madre						
Alelo	1	2	3	4	19	20
Descripción del Alelo	Producto 776	Producto 12	Producto 591	Producto 138	Producto 952	Producto 158

Figura 30. Cruce de alelos

Este proceso se repite hasta generar el total de los 20 alelos del hijo.

g. Se generan **un** hijo.
h. Se aplica la estrategia de reducción

Se Vuelve al paso 7. Hasta que se estabilice.

3.4. Análisis de Complejidad de EKCEAD

Asumamos valores muy grandes para $P_M,\ P_C,\ NumIter,\ MaxIter$:

$$P_M = P_C = NumIter = MaxIter = n \quad \textbf{(75)}$$

Con *n* muy grande al igual que el resto de las variables.

Analizaremos la complejidad, se hará en el peor de los casos. En el algoritmo 3, el marco básico de trabajo (framework) de EKCEAD, las líneas del 1, 2, 4, 6, 7 y 17 suma 6 por las asignaciones realizadas y las líneas 3 y 5 son sub-ecuaciones a calcular. Las líneas 10 y 12, son sub-ecuaciones a calcular. Pero las líneas 8, 9, 11, 14 y 15 se encuentran dentro de una estructura cíclica (línea 7) lo cual generara una sumatorias anidadas y están distribuidas de la siguiente manera.

Las líneas 7 al 15 generan la siguiente expresión matemática:

$$\sum_{i=0}^{n}\left(4 + T_{MUTACION}(n) + T_{CRUCE}(n) + T_{CALCULARESTABILIDAD}(n)\right) \quad \textbf{(76)}$$

Así pues, si se une las ecuaciones (76) y le sumamos 6, la ecuación del análisis inicial será:

$$T_{EKCEAD}(n) = 6 + T_{CARGARCORRELACIONES}(n) + T_{GENERARPOBLACIONINICIAL}(n) + \sum_{i=0}^{n}\left(4 + T_{MUTACION}(n) + T_{CRUCE}(n) + T_{CALCULARESTABILIDAD}(n)\right) \quad \textbf{(77)}$$

Para resolver la ecuación (77) se calculará cada una de las funciones $T_{CARGARCORRELACIONES}(n)$, $T_{MUTACION}(n)$, $T_{CRUCE}(n)$, $T_{GENERARPOBLACIONINICIAL}(n)$ y $T_{CALCULARESTABILIDAD}(n)$de la siguiente manera: En el algoritmo 2 se observa que la ecuación$T_{CARGARCORRELACIONES}(n)$ tiene estructuras cíclicas cíclicas y varias asignaciones, por tanto, su valor es:

$$T_{CARGARCORRELACIONES}(n) = 8 + \sum_{i=1}^{n}\left(1 + \sum_{j=1}^{n}(3)\right) \quad \textbf{(78)}$$

Resolviendo la ecuación (78) se obtiene:

$$T_{CARGARCORRELACIONES}(n) = 3n^2 + n + 8 \quad \textbf{(79)}$$

Por otra parte en el algoritmo 3, en la función $T_{APTITUD}(n)$, las líneas 1, 2 y 10 son asignaciones por tanto suman 3 pero las líneas 3 y 4 son 2 ciclos repetitivos que generan una sumas anidadas y las líneas 5 es una asignación que suman una unidad y las líneas 6 y 7 son sub-ecuaciones que deben resolverse al designare $T_{BUSCARFILA}(n)$, por tanto su valor es:

$$T_{APTITUD}(n) = 3 + \sum_{i=1}^{n}\left(\sum_{j=1}^{n}(2T_{BUSCARFILA}(n) + 1)\right) \quad \textbf{(80)}$$

Para resolver a (80) se debe determinar el valor de función $T_{BUSCARFILA}(n)$, representada en el algoritmo 4 y luego resolverla. En la función $T_{BUSCARFILA}(n)$ se observa que las líneas 1 y 7 son asignaciones que suman 2 unidades, además se observa que las líneas 2 son estructuras cíclicas que generan una suma de la líneas 3 y 4, lo cual sumarian 2 unidades dentro de éste.

Por tanto, su valor viene dado por:

$$T_{BUSCARFILA}(n) = 2 + \sum_{i=1}^{n}(2) \quad \textbf{(81)}$$

Resolviendo la ecuación (81) se obtiene:

$$T_{BUSCARFILA}(n) = 2n + 2 \quad \textbf{(82)}$$

Remplazando la ecuación (82) en la ecuación (80), se obtiene:

$$T_{APTITUD}(n) = 4n^3 + 5n^2 + 3 \quad \textbf{(83)}$$

De igual forma, se observar en el algoritmo 5, tiene estructuras cíclicas y varias asignaciones, por tanto, su valor es:

$$T_{GENERARPOBLACIONINICIAL}(n) = \sum_{i=1}^{n}\left(3 + \sum_{j=0}^{n-1}\left(5 + \sum_{k=1}^{j}(2)\right)\right) \quad \textbf{(84)}$$

Resolviendo (84) se obtiene:

$$T_{GENERARPOBLACIONINICIAL}(n) = 2n^3 + 2n^2 + 3n + 1 \quad \textbf{(85)}$$

Por otra parte, el algoritmo 6 tiene estructuras cíclicas y varias asignaciones, por tanto, su valor es:

$$T_{MUTACION}(n) = 2 + 2T_{SELECCIONAR}(n) + \sum_{i=1}^{n}\left(3 + \sum_{j=1}^{n}(3)\right) \quad \textbf{(86)}$$

Para resolver (86) se debe determinar el valor de función $T_{SELECCIONAR}(n)$ representada en el algoritmo 8 y luego resolverla. En la función $T_{SELECCIONAR}(n)$ se observa que tiene estructuras cíclicas y varias asignaciones, por tanto su valor es:

$$T_{SELECCIONAR}(n) = 7 + 2\left[2 + \sum_{i=1}^{n}\left(2 + T_{APTITUD}(n)\right) + \sum_{i=1}^{n}\left(5 + T_{APTITUD}(n) + \sum_{i=1}^{n}(3)\right)\right] \quad \textbf{(87)}$$

Reemplazando (83) en (87) se obtiene:

$$T_{SELECCIONAR}(n) = 16n^4 + 20n^3 + 32n + 11 \quad \textbf{(88)}$$

Remplazando (88) y resolviendo las sumatorias y efectuando operaciones de simplificación de (86) se obtiene:

$$T_{MUTACION}(n) = 32n^4 + 40n^3 + 64n + 26 \quad \textbf{(89)}$$

De igual forma, se observa en el algoritmo 7, las estructuras cíclicas y varias asignaciones por, por tanto, su valor es:

$$T_{CRUCE}(n) = 10 + T_{SELECCIONAR}(n) + \sum_{i=1}^{n}\left(T_{SELECCIONAR}(n)\right) + 2\sum_{i=1}^{n}\left(2 + T_{APTITUD}(n)\right) + \sum_{i=1}^{n}\left(5 + \sum_{j=1}^{n}(3) + 2\sum_{j=p+1}^{n}\left(3 + \sum_{j=1}^{n}\left(3 + \sum_{k=1}^{i}(3)\right)\right)\right) \quad \textbf{(90)}$$

Para resolver (90) se debe remplazar el valor de la función $T_{SELECCIONAR}(n)$ y $T_{APTITUD}(n)$ Representada en las ecuaciones (83) y (84). Por tanto, su valor es:

$$T_{CRUCE}(n) = 48n^5 + 76n^4 + 29n^3 + 106n^2 + 70n + 19 \quad \textbf{(91)}$$

Terminando el cálculo de las funciones anteriores solo nos resta determinar el valor de la función de $T_{CALCULARESTABILIDAD}(n)$ se observa en el algoritmo 9 algunos ciclos y asignaciones, por lo tanto su valor es:

$$T_{CALCULARESTABILIDAD}(n) = 6 + 2\sum_{i=2}^{30}(1) + \sum_{i=1}^{30}(1) + \sum_{i=1}^{n}\left(T_{APTITUD}(n)\right) \quad \textbf{(92)}$$

Para resolver (92) se debe remplazar el valor de la aptitud $T_{APTITUD}(n)$ representada en las ecuaciones (83) por lo tanto su valor es:

$$T_{CALCULARESTABILIDAD}(n) = 4n^4 + 5n^3 + 33n + 64 \quad \textbf{(93)}$$

Conociendo el valor de las ecuaciones (79), (85), (88), (90) y (92), se remplazarán en la ecuación (77) efectuando operaciones de simplificación de términos y resolviendo las sumas se obtiene:

$$T_{EKCEAD}(n) = 48n^6 + 160n^5 + 186n^4 + 185n^3 + 278n^2 + 279n + 116 \quad (94)$$

Organizando por coeficientes, la ecuación de tiempo en el peor de los casos para EKCEAD es:

$$T_p(n) = 48n^6 + 160n^5 + 186n^4 + 185n^3 + 278n^2 + 279n + 116 \quad (95)$$

Por tanto, la cota superior del algoritmo es igual a:

$$O\left(T_p(n)\right) = max\{48n^6, 160n^5, 186n^4, 185n^3, 278n^2, 279n, 116\} = O(n^6) \quad (96)$$

A partir de (115), se puede obtener la cota inferior de EKCEAD, el análisis es simple:

$$\begin{aligned} T_p(n) &= 48n^6 + 160n^5 + 186n^4 + 185n^3 + 278n^2 + 279n + 116 \\ &\geq 160n^5 + 186n^4 + 185n^3 + 278n^2 + 279n + 116 \\ &\geq 186n^4 + 185n^3 + 278n^2 + 279n + 116 \\ &\geq 185n^3 + 278n^2 + 279n + 116 \\ &\geq 278n^2 + 279n + 116 \qquad \geq 279n + 116 \end{aligned} \quad (97)$$

Por lo tanto, de (97) se concluye que:

$$\Omega\left(T_p(n)\right) = \Omega(279n + 116) \quad (98)$$

4. **Capítulo IV**

4.1. Estudio experimental

En esta sección se presenta la metodología del trabajo de entrenamiento del SCBRDs, como también el conjunto de prueba. Por otra parte, se tomó como software para validación, verificación y medición a una implementación propia de los métodos expuestos en Visual Studio.Net versión 2019 COMMUNITY y como lenguaje de desarrollo Visual Basic.Net.

4.2. Hipótesis

En este numeral se presentan los tipos de hipótesis formuladas y el respectivo diseño metodológico de la presente investigación.

4.2.1. Variables de investigación.

En el proceso de medición de los datos y del modelo propuesto, es necesario enunciar las variables objeto de estudio.

Calidad de respuesta: teniendo en cuenta las siguientes variables.

Precisión: proximidad de las soluciones con el valor optimo, para cada función de prueba que arroje el algoritmo de comparación.

Interpretabilidad: Análisis de la compacticidad (Análisis de comprensibilidad e interpretación de los datos).

4.2.2. Tipos de hipótesis a formular.

*H*i: “Los SCBRDs que utilizan autómatas finitos en su estructura obtienen mejores resultados de predicción, mostrando un buen equilibrio entre la interpretabilidad y la precisión de conocimiento extraído”.

*H*0: “No existe diferencia estadística significativa entre la calidad de predicción aportada por el nuevo algoritmo propuesto y el obtenido a través de los métodos evolutivos más referenciados”.

*H*1: “Existe diferencia estadística significativa de calidad de respuesta aportada por el algoritmo propuesto, en comparación con los métodos evolutivos más referenciados”.

4.3. Tipo de investigación.

En el presente estudio establece un nuevo algoritmo meta heurístico de optimización combinatoria enmarcado en un problema de minería de datos. La investigación se enmarcará en un enfoque netamente CUANTITATIVO, por cuanto el nuevo método a evaluar supone la relación entre variables y amerita el uso de herramientas estadísticas que permitan establecer la existencia de diferencia significativa entre los valores de salida asociados a la variable independiente.

El diseño de investigación será investigación EXPERIMENTAL, partiendo de la acepción la cual refiere si a un estudio en el que se manipulan intencionalmente una o más variables independientes (supuestas causas-antecedentes) para las consecuencias tiene sobre una o más variables (supuestos efectos-causantes), dentro de una situación controlada, la investigación se considera de tipo experimental (Sampieri, Collado, & Baptista, 2010).

4.4. Metodología.

4.5. Diseño experimental.

Teniendo en cuenta los objetivos planteados inicialmente, y con los recursos mencionados anteriormente, la metodología de trabajo fue dividida de la siguiente forma:

1. Estudio de los datos disponibles.
2. Escogencia de los criterios de comparación entre las predicciones.
3. Selección de las técnicas de predicción.
4. Comparación de los resultados de EKCEAD con los algoritmos de entrenamientos citados en la literatura seleccionados en el paso 3.

4.5.1. Estudio de los datos disponibles.

4.5.1.1. Conjunto de datos.

La propuesta se analizó mediante pruebas hechas a la herramienta de minería de datos desarrollada se realizaron con una base de datos de la compañía FoodMart, dicho conjunto de datos consta de diferentes transacciones realizadas por clientes de una cadena de supermercados en los años 1997, 1998. Este conjunto de datos se ha utilizado en diferentes estudios (Vidya & Nedunchezhian, 2011), (Monteserin & Armentano, 2018), (Duong, Fournier-Viger, Ramampiaro, Nørvåg, & Dam, 2018), (Alhusaini et al., 2019), ofrece información sobre ventas, productos y promociones, entre otras características. El conjunto de datos incluye 251,357 transacciones que representan compras de 1559 productos por 8736 clientes. Ver apéndice A.

Estas pruebas fueron de calidad de respuesta teniendo en cuenta las siguientes variables.

- **Precisión**: Proximidad de las soluciones con el valor optimo, para cada función de prueba que arroje el algoritmo de comparación (A. Gokil & S. Rajalakshmi, 2014).
- **Interpretabilidad comprensibilidad:** Análisis de la compacticidad e interpretación de los datos (J. Zhang, Wang, & Feng, 2013).

Por otra parte, es necesario señalar que se utilizó la técnica de validación cruzada con 10 particiones. El conjunto original de ejemplos D se divide en D_j, $J = \{1, \dots, 10\}$ conjuntos disjuntos de igual tamaño. Por lo tanto se construyeron 10 conjuntos de entrenamiento TR_i, $i = \{1, \dots, 10\}$ y sus complementarios 10 conjuntos de prueba TS_i, según las siguientes ecuaciones:

$$TR_i = U_{j \in J}\, D_j \qquad \textbf{(73)}$$

$$J = {}^{j}/_{1} \leq j \leq (i-1)\ y\ (i+1) \leq j \leq 10$$

$$TS_i = \frac{D}{TR_i} \quad \textbf{(74)}$$

Para cada conjunto de datos se mostrarán los resultados en tales iteraciones ver Tabla 9

Tabla 9.Iteraciones EKCEAD

Iteraciones	Confianza	EKCEAD
		Reglas Generadas
1	20	94
2	30	90
3	40	82
4	50	78
5	60	74
6	70	76

4.5.1.2. Escogencia de los criterios de comparación entre las predicciones test no paramétricos.

En este estudio experimental se utilizó un estudio de validación cruzada con 10 particiones para estimar el error del clasificador. Con el fin de entrenar el clasificador con un conjunto de ejemplos independiente del conjunto de prueba para cada una de las particiones y a la vez permite probar el clasificador con todos los ejemplos (Demšar. J., 2006) . Posteriormente, se realizó un test no paramétrico, estos test no paramétricos pueden aplicarse tanto a porcentajes de clasificación como a ratios de error. Para analizar los resultados experimentales obtenidos se utilizó el test de Friedman (F. J. Berlanga, Rivera, del Jesus, & Herrera, 2010), que consiste en detectar diferencias significativas entre algoritmos.

4.5.1.3. Selección de las técnicas de predicción.

El conjunto de datos de pruebas, se compara la metaheurística EKCEAD con los siguientes algoritmos de aprendizaje:

1. A priori: El algoritmo a priori realiza la siguiente secuencia:

   Primero busca todos los conjuntos frecuentes unitarios contando sus ocurrencias directamente en la base de datos), estos se mezclan para formar los conjuntos de ítems candidatos de dos elementos y se seleccionan entre ellos los más frecuentes. Considerando la propiedad de los productos de ítems frecuentes, se vuelven a mezclar estos últimos y se seleccionan los más frecuentes. Así sucesivamente se repite el proceso hasta que en una iteración no se obtengan conjuntos frecuentes (Cuervo & Martínez, 2009).

   De esta forma el algoritmo asume un orden entre los ítem y utiliza las siguientes características: Ck, para el conjunto de datos de k-Itemsets, Fk, para el conjunto de frecuentes de k-itemsets, y asociado a cada itemset se encuentra el campo count para almacenar el soporte de dicho itemset (Romero & Niz, 2013).

2. FP- Growth: El algoritmo está basado en una representación de árbol de prefijos en una base de datos de transacciones denominada frequente Pattern Tree. La idea básica del algoritmo puede ser descrita como un esquema de eliminación recursiva, como primer paso de pre-procesamiento se borran todos los ítems de las transacciones que no son frecuentes individualmente o no aparece en el mínimo soporte de transacciones, luego se seleccionan todas las transacciones que contienen al menos un item frecuente, se realiza la recursividad hasta obtener una base de datos reducida. Al retorno se remueven los ítems procesados de la base de datos de transacciones en la memoria y se empieza otra vez, y así con el siguiente ítem frecuente. Finalmente los ítem en cada transacción son almacenados y ordenados descendentemente su frecuencia en la base de datos (Cuervo & Martínez, 2009), (Ranjan & Sharma, 2019).

4.5.1.4. Análisis comparativo con otros métodos de aprendizaje.

En esta sección se analizan por medio de test estadístico la precisión y la interpretabilidad de EKCEAD, comparando los resultados obtenidos con los resultados de las otras propuestas de aprendizaje de SCBRDs encontradas en la literatura especializada.

4.5.2. Análisis de precisión (Berlanga, 2010)

Los resultados de precisión obtenidos para el conjunto de datos y métodos presentado en este estudio experimental se muestran en la Tabla 10. El estudio estadístico se ha llevado a cabo considerando los niveles de confianza obtenido por los diferentes métodos.

Tabla 10.
Resultados de precisión de EKCEAD y otros métodos de aprendizaje de SCBRDs

Iteraciones	Confianza	EKCEAD	A PRIORI	FP GROWTH
		Reglas Generadas	Reglas Generadas	Reglas Generadas
1	20	94	80	85
2	30	90	75	80
3	40	82	70	75
4	50	78	65	72
5	60	74	60	65
6	70	76	53	60

Nota: Como puede observarse en la tabla los resultados de los algoritmos A priori y FP-Growth tomados de (Vidya & Nedunchezhian, 2011).

En la Figura 31 se muestran los valores de los rankings obtenidos por cada algoritmo mediante el test de Friedman. (Ver anexo 1) Cada columna ejemplifica el ranking medio obtenido por algoritmo.

Si un algoritmo obtiene los rankings 3+3+3+3+3+3 sobre 6 conjuntos de datos, su ranking medio será $\frac{3+3+3+3+3+3}{6} = \frac{18}{6} = 3$, la interpretación esta considera por la altura de la columna, el cual es proporcional al ranking, cuanto más bajo sea la columna mejor es su algoritmo asociado.

En este caso la prueba determina que el mejor algoritmo es a priori, y el EKCEAD presenta el peor ranking asociado.

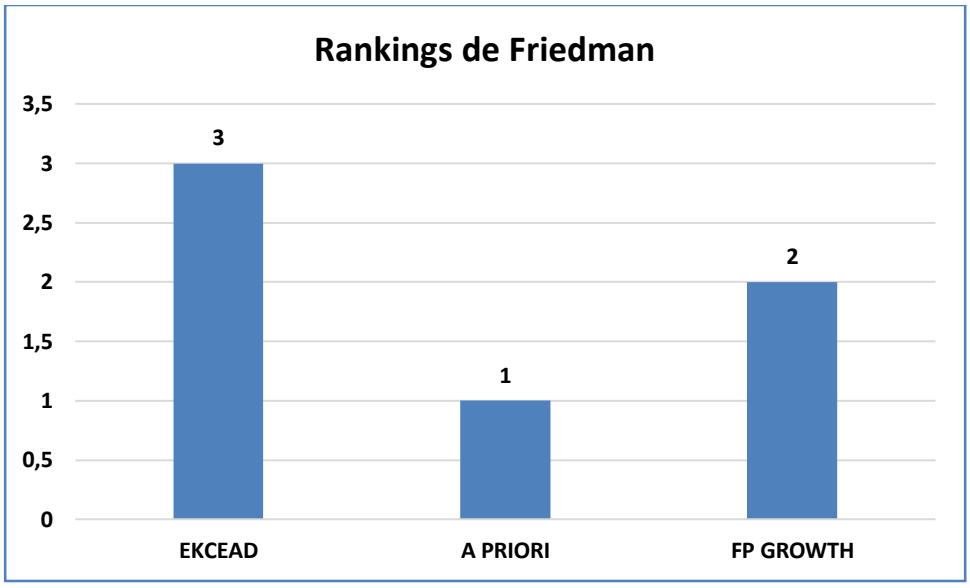


Figura 31. Rankings de Friedman (Precisión)

Para contrastar los métodos relacionados se realizó el test de Friedman con los valores de confianza utilizando como herramienta Statgraphics centurión vs 18 (con un nivel de significancia $\alpha = 0.05$) para determinar si existen diferencias estadísticas entre los métodos. En la Tabla 11 se muestran los resultados del test.

Tabla 11.Estadisticos y valores críticos para test de Friedman Confianza (precisión), $\alpha = 0.05$

Estadístico de Friedman	Valor p	Hipótesis
12	0,00247875	rechazo

Puesto que el valor-P es menor o igual que 0,05, existe una diferencia estadísticamente significativa entre las medianas con un nivel del 95,0% de confianza. Por consiguiente, se rechaza

la hipótesis nula. Luego hay diferencias significativas entre al menos un par de los dos algoritmos referenciados en la literatura y el propuesto en esta investigación.

Tabla 122.Ranking de Friedman

	Tamaño de Muestra	*Rango Promedio*
A PRIORI	6	1,0
EKCEAD	6	3,0
FP GROWTH	6	2,0

Test statistic = 12,0 Valor-P =0,00247875

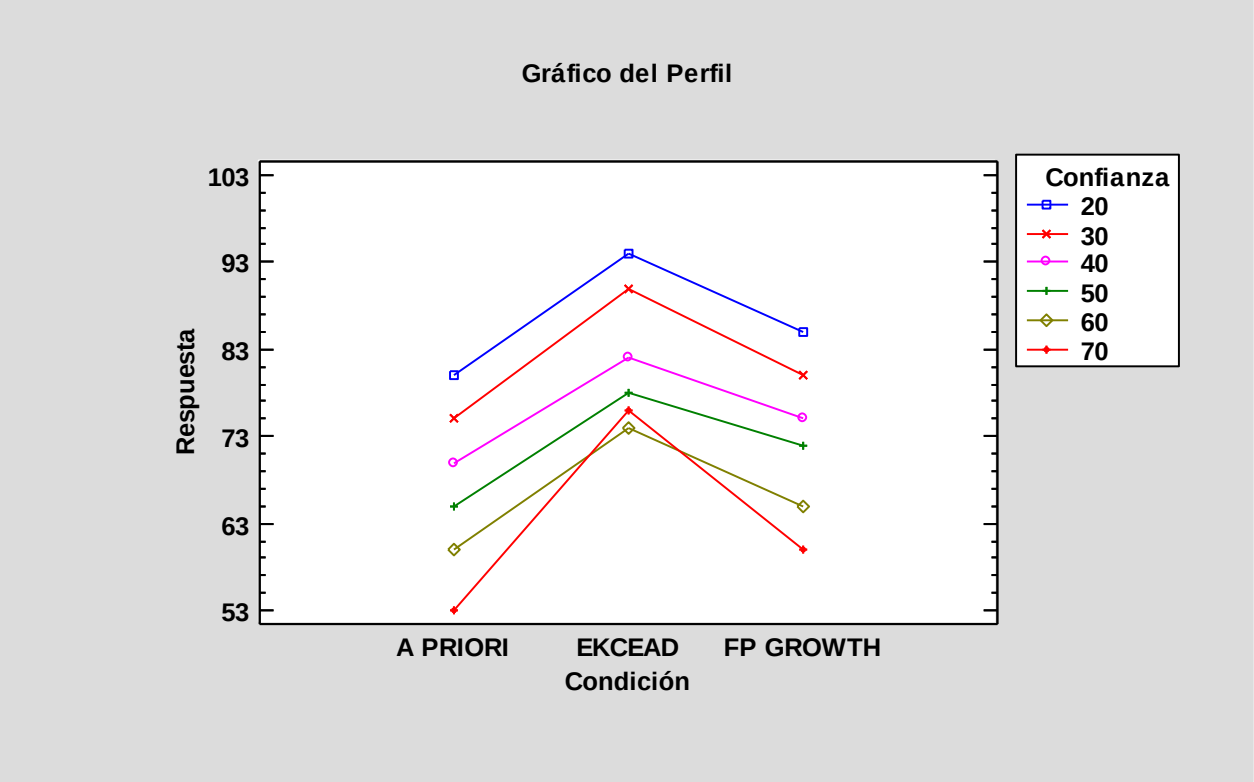


Figura 32. Rankings de Friedman (Confianza)

Existen diferencias significativas entre Ekcead, Apriori, FP-Growth. Dado que, se encontró que EKCEAD cuenta con mayor número de reglas que los dos algoritmos (Apriori; FP-Growth) a un mismo nivel de confianza.

Puesto que, un modelo con menos reglas es más compacto, pero podría tener un menor nivel de cobertura, por tanto, mayor precisión. Sin embargo, un modelo con más reglas tiene mayor rango

de predicción, por tanto, se tiene mayor aplicabilidad, es decir, para el caso de estudio la cesta de compra hay artículos que se compran pocas veces, pero están altamente relacionados.

Es necesario aclarar que el test de Friedman, se limita a encontrar la existencia o no de diferencias en todo el conjunto de resultado, con ello, se puede establecer una ordenación entre algoritmos e incluso medir las diferencias entre ellos (en este ejemplo la precisión de todos los algoritmos estudiados), se concluye que una propuesta es mejor que la otra cuando la hipótesis es rechazada (S. Garcia., J .Derrac., F. Herrera., & D. Molina, 2012).

4.5.3. Análisis de la comprensibilidad (Berlanga, 2010)

La comprensibilidad de una regla puede ser medida por el número de atributos involucrados en la regla (J. Zhang et al., 2013). El IC, es el índice de comprensibilidad e interpretabilidad de una base de reglas. Se cuantifica por la siguiente expresión:

$$IC = 1 - \frac{n}{N}$$

Donde n y N son, respectivamente, el número de atributos en parte antecedente y en todo el conjunto de datos.

La comprensibilidad de una regla intenta aumentar la legibilidad acortando la longitud de una regla de asociación. Es decir, si el número de condiciones involucradas en la parte antecedente es menor, la regla es más comprensible, esto es, cuánto más bajo este el índice, mejor es su comprensibilidad. El índice de comprensibilidad del método propuesto se muestra en la Tabla 13.

Tabla 13. Estadístico comprensibilidad EKCEAD

Comprensibilidad	Resultado	Porcentaje
$IC = 1 - \frac{n}{N}$	0,2	20%

Los valores del índice de compacticidad de los algoritmos Apriori y FP-Growth, no hay evidencias de los resultados en cuanto a la comprensibilidad de los métodos utilizados por los autores de la literatura en referencia. Por tal motivo no se llevó a cabo el análisis estadístico entre los métodos e indicar si existen o no diferencias significativas.

5. **Resultados obtenidos y discusión**

En las diferentes ejecuciones del código se pudo verificar que la implementación de un autómata finito determinístico representado en este caso por los pasos a seguir por el cálculo de correlaciones, aptitudes, el mantenimiento del grupo elite contribuyen a reducir los tiempos de ejecución necesarios para construir un conjunto de reglas. No obstante, persiste la complicación inherente al costo computacional relativo a la explosión combinatoria, esto es, entre más artículos y días deben ser procesados se hace necesario utilizar un mayor número de agentes de búsqueda para aprovechar el paralelismo (estrategia de enjambres) pero la construcción de más agentes de búsqueda exige más recursos computacionales.

Se realiza un estudio comparativo utilizando el test de Friedman para demostrar las hipótesis planteadas al inicio de la investigación y se encuentra evidencias significativas que apoya el planteamiento que el uso de autómatas disminuye los tiempos de ejecución necesarios para la construcción del conjunto de reglas.

***H*i: "Los SCBRDs que utilizan autómatas finitos en su estructura obtienen mejores resultados de predicción, mostrando un buen equilibrio entre la interpretabilidad y la precisión de conocimiento extraído".**

El uso de autómatas para el proceso de extracción e interpretación de reglas difusas aun cuando mejora en cierta forma el proceso de interpretabilidad de las reglas, no incide de forma sensible en la precisión de las mismas. Parte de la investigación comprueba que al ser el autómata un método principalmente determinístico, y la estrategia evolutiva un método primordialmente heurístico, no se produce una integración completa entre las formas de operar de dichos métodos.

***H*0: "No existe diferencia estadística significativa entre la calidad de predicción aportada por el nuevo algoritmo propuesto y el obtenido a través de los métodos evolutivos más referenciados".**

Se ejecutó el algoritmo EKCEAD con diversos parámetros ajustados y al principio existía una diferencia significativa en los tiempos de ejecución (llegó a ser de aproximadamente una hora por cada corrida); posterior al proceso de optimización y como resultado de comparar los resultados se observa que la cantidad y calidad de las reglas difusas generadas es similar a la de los métodos referenciados en la literatura y que incluso en algunos casos, con representaciones cromosómicas de más de 20 genes resulta ser de mejor calidad.

No obstante, se debe enfatizar que en la medida que la cantidad de cromosomas y genes aumenta, se requiere de mucho más tiempo para poder alcanzar la estabilidad poblacional, razón por la cual se configuró un sistema de ejecución con refinamiento donde se utiliza un

número máximo de iteraciones para poder detener el método y utilizar la salida lograda como una entrada para una nueva ejecución. Ver figura 33.

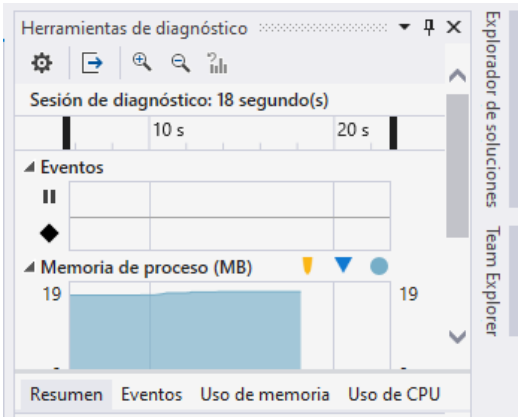


Figura 33. Tiempo de ejecución y uso de memoria

***H*1: "Existe diferencia estadística significativa de calidad de respuesta aportada por el algoritmo propuesto, en comparación con los métodos evolutivos más referenciados".**

Existen diferencias estadísticamente significativas, a modo de evidencia se ilustran los resultados consignados en el test de Friedman, ver sesión 4.4.2.

Se formularon las siguientes **preguntas de investigación**.

¿Cómo combinar la teoría de autómatas, el cómputo evolutivo y la lógica difusa para descubrir conocimiento de interés dentro de los datos que permita describir y representar el conocimiento de forma comprensible, mediante reglas difusas?

Para descubrir conocimiento de interés dentro de los datos se planteó una. La matriz de correlaciones que constituye la lógica difusa. Dada la matriz se aplica la función objetivo que posibilita el cómputo evolutivo y el autómata guía el proceso incluyendo la interpretación de la regla. Ver Figura 34.

¿La presencia de autómata finito en el aprendizaje un SCBRDs incide en los métodos de descubrimiento de conocimiento, de modo que no se produzcan repeticiones en el proceso aprendizaje y búsqueda de soluciones optimas?

El autómata que se utiliza en el procedimiento no incide en el método para el descubrimiento de conocimiento, el algoritmo genético mediante cruces y mutaciones produjeron reglas repetidas, es decir, se obtuvieron reglas que ya se encontraban en la base de datos.

ANTS: Mostrar correlaciones [1.5.0]

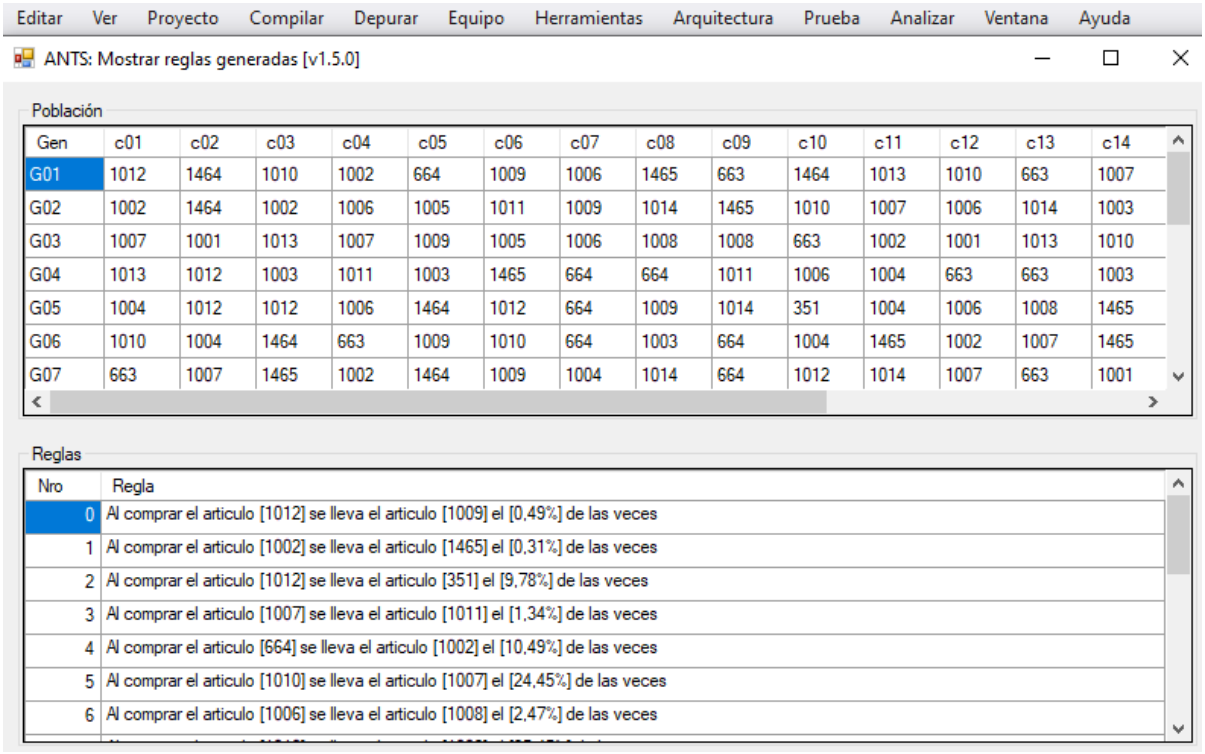
Producto	351	664	663	1003	1001	1008	1010	1006	1004	1007	1002	1013	1009
351	1,000	0,706	0,533	0,580	0,290	0,302	0,775	0,014	0,761	0,814	0,709	0,045	0,414
664	0,706	1,000	0,056	0,950	0,364	0,525	0,767	0,054	0,592	0,469	0,298	0,623	0,648
663	0,533	0,056	1,000	0,986	0,911	0,227	0,695	0,980	0,244	0,534	0,106	0,999	0,676
1003	0,580	0,950	0,986	1,000	0,284	0,046	0,296	0,382	0,301	0,949	0,980	0,401	0,278
1001	0,290	0,364	0,911	0,284	1,000	0,713	0,326	0,633	0,208	0,186	0,583	0,081	0,458
1008	0,302	0,525	0,227	0,046	0,713	1,000	0,919	0,632	0,628	0,428	0,098	0,561	0,694
1010	0,775	0,767	0,695	0,296	0,326	0,919	1,000	0,430	0,678	0,502	0,514	0,463	0,353
1006	0,014	0,054	0,980	0,382	0,633	0,632	0,430	1,000	0,061	0,390	0,365	0,490	0,156
1004	0,761	0,592	0,244	0,301	0,208	0,628	0,678	0,061	1,000	0,939	0,654	0,506	0,390
1007	0,814	0,469	0,534	0,949	0,186	0,428	0,502	0,390	0,939	1,000	0,833	0,019	0,210
1002	0,709	0,298	0,106	0,980	0,583	0,098	0,514	0,365	0,654	0,833	1,000	0,537	0,657
1013	0,045	0,623	0,999	0,401	0,081	0,561	0,463	0,490	0,506	0,019	0,537	1,000	0,454
1009	0,414	0,648	0,676	0,278	0,458	0,694	0,353	0,156	0,390	0,210	0,657	0,454	1,000
1014	0,863	0,264	0,016	0,160	0,906	0,914	0,405	0,474	0,107	0,074	0,544	0,357	0,090
1012	0,790	0,279	0,575	0,163	0,261	0,835	0,270	0,257	0,784	0,105	0,827	0,150	0,758
1011	0,374	0,830	0,100	0,647	0,785	0,023	0,056	0,629	0,460	0,332	0,082	0,704	0,402
1005	0,962	0,825	0,103	0,410	0,379	0,543	0,244	0,542	0,754	0,128	0,192	0,929	0,462
1464	0,871	0,589	0,799	0,413	0,290	0,916	0,979	0,156	0,596	0,000	0,679	0,530	0,492
1465													

Figura 334. Matriz de correlación EKCEAD

¿Cómo validar la técnica utilizada, de tal manera que se verifique su aplicabilidad al problema de investigación?

La forma como se verifico la técnica utilizada con respecto a los factores calidad de la regla: Precisión e interpretabilidad de la regla, fue que se tomaron los datos del año 1997

para poder generar las reglas y se utilizó como contraste los datos del año 1998 para verificar que tan bien la regla predecía o que tan bien la regla se ajustaba a esos nuevos datos. Por ello se calcularon los niveles de soporte y confianza, para estimar las veces que una regla haya sido aplicada predice acertadamente cual es el siguiente artículo que se va a encontrar en el carrito de compra. Ver figura 35.



Gen	c01	c02	c03	c04	c05	c06	c07	c08	c09	c10	c11	c12	c13	c14
G01	1012	1464	1010	1002	664	1009	1006	1465	663	1464	1013	1010	663	1007
G02	1002	1464	1002	1006	1005	1011	1009	1014	1465	1010	1007	1006	1014	1003
G03	1007	1001	1013	1007	1009	1005	1006	1008	1008	663	1002	1001	1013	1010
G04	1013	1012	1003	1011	1003	1465	664	664	1011	1006	1004	663	663	1003
G05	1004	1012	1012	1006	1464	1012	664	1009	1014	351	1004	1006	1008	1465
G06	1010	1004	1464	663	1009	1010	664	1003	664	1004	1465	1002	1007	1465
G07	663	1007	1465	1002	1464	1009	1004	1014	664	1012	1014	1007	663	1001

Figura 35. Porcentaje de acierto de las reglas generadas

¿Cuál es el nivel de calidad de respuesta que proporciona el nuevo algoritmo meta heurístico propuesto?

El nivel de calidad de respuesta depende de la longitud (representación) que se le aplica al individuo, en el proceso de investigación se realizó el aprendizaje del algoritmo con tres

longitudes 10, 20, 30, observando que entre más corto es el individuo o sea para individuos menos características las reglas llegan a ser más exactas en cuanto a la predicción del comportamiento poblacional.

¿Representan esta calidad de respuesta alguna mejoría en comparación con las metas heurísticas más referenciadas en la literatura?

Hasta el momento en la investigación no se encontró ninguna mejoría, incluso se hizo una optimización del método para poderlo ajustar en cuanto al tiempo de ejecución. Sería interesante a futuro con un supercomputador o con un computador de mejores características ejecutar el método no solo 20 0 30 veces sino muchas más veces (ver apéndice B), para poder asegurar con total certeza y con mayores datos de análisis determinar si es realmente superior o no, porque con las 30 veces que se ejecutó lo que podemos estar observando son picos mínimos locales y no una tendencia generalizada, probada en una ejecución masiva del método.

¿Cuáles son los valores de los parámetros que optimizan el funcionamiento del algoritmo propuesto?

Los valores de los parámetros son los siguientes:

Número máximo de iteraciones =1000

Tolerancia para alcanzar estabilidad = 2%

Cantidad de genes de un individuo sujeto a modificaciones = 30,20, 10

Cantidad de individuos en la población sujeto a modificaciones =100, 50, 20

Tamaño de la población elite = 10.000

Probabilidad de cruce= 0.2 y mutación = 0.8

Tolerancia= 0.05

Todos estos parámetros se establecieron como constantes, pero con la posibilidad de realizar modificaciones.

6. **Conclusiones**

En este trabajo, se propuso un método basado en cómputo evolutivo para el aprendizaje de un SCBRDs para problemas con alta dimensionalidad denominado EKCEAD sus principales características son las siguientes:

- Representa la extracción de conocimiento mediante reglas difusas con notación DNF.
- Utiliza una gramática libre de contexto para aprender las reglas difusas DNF, lo cual permite la ausencia de algunas variables de entada en las reglas, esto nos permite obtener reglas con pocas variables y etiquetas lingüísticas en sus antecedentes.
- Utiliza como enfoque de codificación de reglas el enfoque de MICHIGAN, el cual codifica una única regla por individuo. La base de reglas la forman todas las reglas (individuos) de la población.
- Introduce un mecanismo para incrementar la diversidad de la población, denominado competición Elitista, el cual hace que las reglas compitan entre sí durante el proceso evolutivo para seleccionar las mejores, permitiendo la obtención de reglas con una alta capacidad de generalización.
- Utiliza estrategias de enjambre como son el paralelismo y la comunicación entre individuos (agentes) que permite al algoritmo encontrar soluciones de alta calidad y moverse por áreas sin exploración del espacio de búsqueda. Usa un proceso formado por capas de niveles iterativos, cada agente genera una regla local obtenida, esta es compartida entre todos los agentes, y a través de esta comunicación se selecciona la solución global.
- Incluye un autómata finito para la representación del espacio factible, que determina el conjunto de reglas que representa la base de reglas, permitiendo guardar cada regla seleccionada, de modo, que no se produzcan repeticiones.

- Un estudio experimental que incluye un conjunto de datos de alta dimensión y otros dos algoritmos de aprendizaje basados en reglas difusas bien conocidos en la literatura especializada, se ha utilizado un método estadístico no paramétricos para comparar y analizar la interpretabilidad y la precisión de los algoritmos.

Las principales conclusiones son las siguientes:

- EKCEAD no supera al resto de los algoritmos con respecto a los resultados de precisión en los datos de prueba. Ha sido demostrado para poder obtener FRBCS con una alta capacidad de generalización para problemas de alta dimensión.
- EKCEAD puede aprender SCBRD en problemas de alta dimensión.
- El autómata no incide favorablemente o de manera no determinante en la eficiencia del método.
- Si bien la estrategia evolutiva utilizada resulta efectiva para generar reglas con cierto grado de precisión; Las dificultades inherentes para definir la función de aptitud y los cálculos previos para determinar la matriz de correlaciones, limitan la aplicabilidad en un entorno sujeto a los conceptos propios de Big Data (el concepto de las tres V, Velocidad, Variedad y Volumen).
- Así mismo es importante resaltar que las reglas generadas incluyen factores de certeza; pero a menos que se ejecuten durante mucho tiempo las estrategias evolutivas al utilizar un fuerte componente de aleatoriedad.
    a) Incluyen la posibilidad de generar reglas repetidas y
    b) Realizan una exploración del espacio se soluciones de forma heurística, lo cual le permite acercarse a la mejor solución, pero no garantiza encontrarla. En el caso de

estudio, se observa que luego de muchas ejecuciones el método suele generar soluciones similares, pero no idénticas.

7. **Limitaciones teórico prácticas de EKCEAD.**

- Los algoritmos genéticos no utilizan de forma intensiva el conocimiento y no cuentan con habilidades de aprendizaje; esto se convierte en una limitante.
- La base de datos con que se trabajó no diferencia entre las compras realizadas el mismo día, esto es un cliente puede realizar más de una compra por día; esta situación afecta de cierta forma el resultado al asumir que todas las compras de un cliente el mismo día fueron realizadas juntas.
- El consumo de memoria y recursos de CPU no es excesivo, pero puede crecer exponencialmente dado que la matriz de correlaciones es cuadrada n*n, donde n es la cantidad de artículos diferentes. Se podría haber experimentado con una técnica de almacenamiento basada en matrices dispersas.
- La determinación de los parámetros óptimos para el método se ve fuertemente afectada por los datos de estudio; por ende, se basa en un ciclo de ensayo y error con un aporte de la experticia del investigador.

8. **Líneas de investigación futuras**

A continuación, se presentan algunas líneas de trabajo relacionado con el tema tratado en esta investigación.

En esta propuesta se ha presentado un modelo evolutivo para abordar el aprendizaje de SCBRDs en problemas que presentan una alta dimensionalidad con respecto al número de variables de entrada. Siendo éste frecuente cuando el problema a solucionar representa un conjunto de reglas para determinar la relación o asociación que existe entre ellas. En este tipo de problemas el modelo evolutivo presenta dificultades al procesar grandes cantidades de reglas, que pueden afectar en gran medida la interpretabilidad y la precisión del conocimiento extraído, para solucionar este problema, la literatura ofrece como aporte el uso de algoritmos de aprendizaje no supervisado de distancias, los cuales mejoran los resultados de los métodos de agrupamiento. Por este motivo, sería interesante realizar una investigación sobre el funcionamiento del modelo evolutivo propuesto junto con el uso de algoritmos no supervisados de aprendizaje de distancias para resolver problemas que presenten alta dimensionalidad.

Con respecto al modelo formal, éste puede seguir adaptándose a nuevas medidas de la función fitness como ejemplo desviación estándar, con el fin de realizar comparaciones de predicción del método con los distintos criterios de la función fitness. De igual modo, se podrían realizar adaptaciones al modelo en cuanto al criterio de las medidas de interés, en esta propuesta se analizaron el soporte y la confianza para discernir entre las reglas más interesantes para cada usuario en el proceso de minería de datos, sin embargo, hay otras como por ejemplo Confianza centrada, Lift, Mínima contradicción, factor de certeza. Que también son utilizadas en el proceso de extracción de conocimiento para medir, cuantificar la utilidad y valor de las reglas de asociación obtenidas en los datos. Usando el factor certeza se consigue una reducción del número de reglas extraídas, de modo, que permite extraer reglas de asociación consideradas muy fuertes.

No obstante, al algoritmo propuesto no se le realizo alteraciones en los parámetros de optimización, por tanto, se propone investigar nuevos ambientes, realizando una serie de ensayos y pruebas en los parámetros de optimización, modificando, por ejemplo, la tolerancia para alcanzar la estabilidad por un valor distinto al 2%, y también el valor de la tolerancia $\neq 0.05$. Para obtener unos resultados más favorables o para confirmar la información ya obtenida, es decir, que permita conocer si los parámetros iniciales son los más adecuados, y así determinar mediante el Ranking de Friedman, si el algoritmo propuesto tiene mayor precisión y comprobar si logra superar a los dos algoritmos presentados en la literatura.

Por último, se recomienda experimentar con el algoritmo en otras bases de datos para probar y validar el modelo.

BIBLIOGRAFIA

A. Gokil, & S. Rajalakshmi. (2014). Weighted Quamtum Particle Swarm Optimization(WQPSO) and PSO algorithm to association Rule Minng and clustering. *Assian Journal of Information Technology*, *13(10)*, 582–587.

Abe. S., & Lan M.-S. (1995). A method for fuzzy rules extraction directly from numerical data and its application to pattern classi- fication. *IEEE Transactions on Fuzzy Systems*, *3(1):*, 18–28.

Abe.S., & Thawonmas, R. (1997). A fuzzy classifier with ellipsoidal regions. *IEEE Transactions on Fuzzy Systems*, *5(3):*, 358–368.

Abonyi. J., Roubos ,J. A., & Szeifert. F. (2003). Data driven genera tion of compact, accurate, and linguistically sound fuzzy classifiers based on a decision-tree initialization. International Journal of Approximate Reasoning. *International Journal of Approximate Reasoning*, *32(1):*, 1–21.

Abonyi J., R. J. A. y S. F. (2003). Data driven genera tion of compact, accurate, and linguistically sound fuzzy classifiers based on a decision-tree initialization. International Journal of Approximate Reasoning. *International Journal of Approximate Reasoning*, *32(1):*, 1–21.

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., & Verkamo, I. (1996). *Fast Discovery of Association Rules, in Advances in Knowladge Discovery and Data Mining* (pp. 307–328). pp. 307–328. California.

Aguilar. L. (2019). *Analisis comparativo en la implementacion de la red neuronal backpropagation usando el metodo de componentes principales y el metodo clasico*. Universidad nacional de piura escuela de posgrado.

Al-Omairi, L., Abawajy, J., Chowdhury, M., & Al-Quraishi, T. (2019). High-Dimensionality Graph Data Reduction Based on Proposing A New Algorithm. *EpiC Series Computing*, *63*, 1--10. https://doi.org/10.29007/h232

Alcalá R., Alcalá-Fdez J., H. F. y O. J. (2007). Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation. *International Journal of Approximate Reasoning*, *44(1):*, 45–64.

Alhajj, R., & Kaya, M. (2008). Multi-objective genetic algorithms based automated clustering for fuzzy association rules mining. *Journal of Intelligent Information Systems*, *31*(3), 243–264. https://doi.org/10.1007/s10844-007-0044-1

Alhusaini, N., Karmoshi, S., Hawbani, A., Jing, L., Alhusaini, A., & Al-sharabi, Y. (2019). LUIM: New Low-Utility Itemset Mining Framework. *IEEE Access*, *7*, 100535–100551. https://doi.org/10.1109/access.2019.2929082

Alvarez, A., Boente, G., & Kudraszow, N. (2019). Robust sieve estimators for functional canonical correlation analysis. *Journal of Multivariate Analysis*, *170*, 46–62. https://doi.org/10.1016/j.jmva.2018.03.003

Asadi, A., Afzali, M., Shojaei, A., & Sulaimani, S. (2012). New binary PSO based method for finding best thresholds in association rule mining. *Life Science Journal*, *9*(4), 260–264.

Bäck, T. (1996). Evolutionary algorithms in theory and practice. *Ox- Ford University Press.*

Bäck, T., Fogel. D, & Michalewicz. Z. (1997). Handbook of evolutionary computation. . . *Oxford University Press.*

Bäck, T., & Schwefel H. (1991). Extended selection mechanisms in genetic algorithms. *En Proc. Fourth International Conference on Genetic Algorithms (ICGA'91),* páginas 2–9. San Diego, EE.UU.

Baker J. E. (1987). *Reducing bias and inefficiency in the selection algorithm. En Grefenstette J. J. (Ed.) Proceedings of the 2nd In- ternational Conference on Genetic Algorithms (ICGA'87),*. páginas 14–21. . Lawrence Erlbaum Associates, Hillsdale, NJ, EE.UU.

Bandyopadhyay, S., Saha, S., Maulik, U., & Deb, K. (2008). A simulated annealing-based multiobjective optimization algorithm: AMOSA. *IEEE Transactions on Evolutionary Computation*, *12*(3), 269–283. https://doi.org/10.1109/TEVC.2007.900837

Berlanga, F. (2010). *Aprendizaje de Sistemas Basados en Reglas difusas compactas y precisas con programacion genetica*. Tesis Doctoral.Universidad de Jaen, Andalucia. España., Andalucia ,España.

Berlanga, F. J., Rivera, A. J., del Jesus, M. J., & Herrera, F. (2010). GP-COACH: Genetic Programming-based learning of COmpact and ACcurate fuzzy rule-based classification systems for High-dimensional problems. *Information Sciences*, *180*(8), 1183–1200. https://doi.org/10.1016/j.ins.2009.12.020

Bhatt R. B. y Gopal M. (2005). On fuzzy-rough sets approach to feature selection. *Pattern Recognition Letters*, *26(7):*, 965–975.

Bouchachia A. Mittermeir R. (2007). Towards incremental fuzzy classifiers. *Soft Computing*, *11(2):*, 193–207.

Brad, L. M., & Goldberg, D. E. (1995). Genetic Algorithms, Tournament Selection, and the Effects of Noise. *Connect*, *9*, 13–13.

Breiman. L. (1984). *Classification and Regression Trees*. Monterey, Wadsworth and Brooks.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and Regresion Tress*.

C., F., Hernández. J., & Ramírez. J. (2001). Learning MDL-guided Decision Trees for Constructor-Based Languages". *In Proceedings of 11th International Conference on Inductive Logic Programming*, 39–50.

C. J. van Rijsbergen. (1979). "Information retrieval". *Butterworths.*

Cagnina, L. (2010). *Optimización Mono y Multiobjetivo a través de una Heurística de Inteligencia Colectiva",*. Universidad Nacional de San Luis, Argentina,.

Calders, T., & Verwer, S. (2010). Three naive Bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, *21*(2), 277–292. https://doi.org/10.1007/s10618-010-0190-x

Cardoso. P, Jesus. M, & Marquez. A. (20003). *MONACO - Multi-Objective Network Optimisation Based on an ACO*. (November).

Carmona, J. (2011). Descubrimiento de subgrupos mediante sistemas difusos evolutivos. *II Jornadas Andaluzas de Informática, Canillas Del Aceituno (España)*, 30-35. Retrieved from http://simidat-web.ujaen.es/~simidat/sites/default/files/publicaciones/220.pdf

Casillas. J., Carse.B., & Bull. L. (2007). Fuzzy-XCS: a michigan ge- netic fuzzy system. *IEEE Transactions on Fuzzy Systems*, *15(4)*, 536–550.

Casillas. J, Cordón. O, del Jesus M., & Herrera. F. (2005). Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction. *IEEE Transactions on Fuzzy Systems*, *13(1):* ., 13–29.

Casillas. J, Cordón. O, Herrera. F, & del Jesus. M. (2001). Genetic feature selection in a fuzzy rule-based classification system learn- ing process for high-dimensional problems. *Information Sciences*, *136(1-4):*, : 135–157.

Casillas. J, Cordón. O, Herrera. F, & Magdalena. L. (2003). Interpretability Issues in Fuzzy Modeling. *Of Studies in Fuzziness and Soft Computing. Springer-Verlag.*, *128*.

Casillas, Cordón. O, Herrera. F., & Magdalena. L. (2003). Accuracy Improvements in Linguistic Fuzzy Modeling,. *In Fuzziness and Soft Computing. Springer-Verlag.*, *129 of Stu*.

Castellano G., Castiello C., F. A. M. y M. C. (2005). Knowledge discovery by a neuro-fuzzy modeling framework. *Fuzzy Sets and Systems*, *149(1):*, 187–207.

Castellano G., Fanelli. A., & Mencar.c. (2002). A neuro-fuzzy network to generate humanunderstandable knowledge from data. *Cognitive Systems Research*, *3(2):*, 125–144.

Castellano G. y Fanelli A. M. (2000). A staged approach for gener- ation and compression of fuzzy classification rules. En Proc. *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2000), San Antonio, USA.*, 42–

47.

Castro, D. (2004). *Teoría de Autómatas, Lenguajes formales Y Gramatica* (Universidad de Alcalá). Retrieved from https://portal.uah.es/portal/page/portal/GP_EPD/PG-MA-ASIG/PG-ASIG-78021/TAB42351/talfg_notes.pdf

Castro P. A. D. y Camargo H. A. (2004). Learning and optimiza- tion of fuzzy rule base by means of self-adaptive genetic algorithm. En Proc. *EEE International Conference on Fuzzy Systems (FUZZ-IEEE 2004) Budapest, Hungary.*, 1037– 1042.

Chaiyaratana, N. (1997). *Recent developments in evolutionary and genetic algorithms: theory and applications*. (4), 270–277. https://doi.org/10.1049/cp:19971192

Chakraborty. D., & Pal N.R. (2004). A neuro-fuzzy scheme for simul- taneous feature selection and fuzzy rule-based classification. *IEEE Transactions on Neural Networks, 15(1):*, 110–123.

Chen M.-Y. y Linkens D. A. (2004). Rule-base self-generation and simplification for data-driven fuzzy models. *Fuzzy Sets and Systems, 142(2):*, 243–265.

Chen, W., Pradhan, B., Li, S., Shahabi, H., Rizeei, H. M., Hou, E., & Wang, S. (2019). Novel Hybrid Integration Approach of Bagging-Based Fisher's Linear Discriminant Function for Groundwater Potential Analysis. *Natural Resources Research*, *28*(4), 1239–1258. https://doi.org/10.1007/s11053-019-09465-w

Chi Z. y Yan H. (1996). ID3-Derived fuzzy rules and optimized defuzzification for handwritten numeral recognition. *IEEE Trans- Actions on Fuzzy Systems*, *4(1):*, 24–31.

Chi Z. Yan H. Pham T. (1996). Fuzzy algorithms with applications to image processing and pattern recognition. *World Scientific.*

Clark, P., & Niblett, T. (1989). The CN2 rule induction algorithm. *Machine Learning*, *3*(4), 261–284. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.3672&rep=rep1&type=pdf

Cohen, W. W. (2014). Fast Effective Rule Induction. In *Machine Learning Proceedings 1995*. https://doi.org/10.1016/b978-1-55860-377-6.50023-2

Combs. W. E., & Andrews. J. (1998). Combinatorial rule explosion eliminated by a fuzzy rule configuration. *IEEE Transactions on Fuzzy Systems*, *6(1):*, 1–11.

Cordón, O., Herrera, F., Hoffmann, F., and Magdalena, L. (2001). Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases. *World Scientific.*

Cordón, O., Herrera, F., Hoffmann, F., and Magdalena, L. (2004). Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases. *World Scientific.*

Cordón. O., del Jesus. M., & Herrera. F. (1999). A proposal on reasoning methods in fuzzy rule-based classification systems. *In- Ternational Journal of Approximate Reasoning*, *20*(21–45).

Cordon. o., Herrera. F, Magdalena. L., & Hoffmann. F. (2004). Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases. In *Fuzzy Sets and Systems* (Vol. 141). https://doi.org/10.1016/S0165-0114(03)00262-8

Cordón.O., del Jesus. M, F., H., & Lozano.M. (1999). MOGUL: A Methodology to Obtain Genetic fuzzy rule-based sys- tems Under the iterative rule Learning approach. *International Journal of Intelligent Systems, 14 (11),* 1123–1153.

Cuervo, R. C. N., & Martínez, L. M. S. (2009). Herramienta software para el análisis de canasta de mercado sin selección de candidatos. *Ingenieria e Investigacion*, *29*(1), 60–68.

De Jong. K.A. (1975). An analysis of the behavior of a class of genetic adaptive systems, 1975. *Univ. Michigan.* https://doi.org/Microfilms Number 76-9381

Deb, K. (2001). Multi objetive optimization using evolutionary algortithms. *Jhon Wiley & Sons.*

Del jesus, M., González, P., & Herrera;Francisco. (2005). Induccion evolutiva multiobjetivo de reglas de descripcion de subgrupos en un problema de marketing. *Actas Del IV Congreso Español Sobre Metaheuristicas, Algoritmosevolutivos y Bioinspirados MAEB2005*, *I 84-9732*, 661–669. Retrieved from http://sci2s.ugr.es/publications/ficheros/deljesus_gonzalez_herrera_CEDI-MAEB05.pdf

Del Jesus, M., González, P., & Herrera, F. (2007). Multiobjective Genetic Algorithm for Extracting Subgroup Discovery Fuzzy Rules. *Intelligence in Multicriteria Decision Maeketing*, (Mcdm), 50–57. Retrieved from http://sci2s.ugr.es/keel/pdf/algorithm/congreso/2007-delJesus-MCDM.pdf

DelJesus, Herrera, Mesonero, & Gonzalez. (2008). Algoritmo Evolutivo de Extracción de Reglas de Asociación aplicado a un Problema de Marketing. *Ministerio de Ciencia y Tecnología*. Retrieved from http://sci2s.ugr.es/keel/pdf/keel/congreso/maeb04-reglas-jaen-granada.pdf

Demšar. J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research, 7*, 1–30.

Derrac, J., Herrera, F., & García, S. (2010). *Un Tutorial Metodológico para hacer Comparaciones Estadísticas con Tests No Paramétricos en Propuestas de Minería de Datos*.

Dietterich. T.G. (1998). aproximate statisticial test for comparing supervised classification learning algorithms. *Neural Commpution, 10 (7)*.

Doerner, K. F., Hartl, R. F., & Reimann, M. (2001). Are {COMPETants} more competent for problem solving? - the case of a routing and scheduling problem. *Genetic and Evolutionary Computation Conference*, 802.

Donoso, Yezid; Fabregat, R. (2007). Multi-Objective Optimization in Computer Networks Using Metaheuristic. *Aurbach Publications New York, Estados Unidos*, 1 – 3.

Duong, Q. H., Fournier-Viger, P., Ramampiaro, H., Nørvåg, K., & Dam, T. L. (2018). Efficient high utility itemset mining using buffered utility-lists. *Applied Intelligence*, *48*(7), 1859–1877. https://doi.org/10.1007/s10489-017-1057-2

E. L. Lawler, Jan Karel Lenstra, A. H. G. Rinnooy Kan, and D. B. S. (1985). *The-Traveling-Salesman-Problem-A-Guided-Tour-of-Combinatorial-Optimization-* (Wiley & C. Series in Discrete Mathematics & Optimization. Wiley Interscience, Eds.).

Eschenauer H.A. (1988). *Multicriteria Optimization Techniques for Highly Accurate Focusing Systems. In: Stadler W. (eds) Multicriteria Optimization in Engineering and in the Sciences. Mathematical Concepts and Methods in Science and Engineering*, (vol 37.; M. Springer, Boston, Ed.).

Fayyad. (1994). Branching on attribute values in decision tree generation". *In Proceedings of the Twelfth National Conference on Artificial Intelligence, 1*, 601-606.

Fayyad, U., Piatetsky-shapiro, G., & Smyth, P. (1996a). From Data Mining to Knowledge Discovery in Databases. *AL MAGAZIN*, *17*(3), 37–54.

Fayyad, U., Piatetsky-shapiro, G., & Smyth, P. (1996b). *The KDD Process for Extracting Useful Knowledge from Volumes of Data*. *39*(11), 27–34.

Ferri. C., Hernández. J., & Ramírez. J. (2002). Learning Decision Trees Using the Area Under the ROC Curve. *In Proceedings of the 19th International Conference on Machine Learning*, 139–146.

Flockhart, I.W. and Radcliffe, N. J. (1995). GA-MINER: Parallel data mining with hierarchical genetic algorithms. *University of Edimburgh, UK*.

Fogel D. B. (1988). *An evolutionary approach to the travelling sales man problem. Biological Cybernetics*. *60(2):*, 139–144.

Freitas, A. A. (1999). On rule interesting ness measures. *Knowledge Based System, 15*, 309–315.

Freitas, A. A. (2002). *Data Mining and Knowledge Discovery with Evolutionary Algorithms.* Springer.

Fuchs, M. (1999). Genetic, Large Populations Are Not Always The Best Choice In Computation, Programming. *Proceedings of the Genetic and Evolutionary Conference (GECCO-99).*, 1033-1038. Orlando, FL,: Morgan-Kauffman,.

Gambardella.L., Taillard. E, & Agazzi. G. (1999). MACS-VRPTW: un sistema de colonias múltiples para problemas de enrutamiento de vehículos con Windows de tiempo. *Cite Ceer*, 73–76.

Garcia. A. L., & Tsang. E. P. (2006). Simplifying Decision Trees Learned by Genetic Programming. *In IEEE Congress on Evolutionary Computation*, (2142–2148).

García, S., Derrac, J., Cano, J. R., & Herrera, F. (2012). Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(3), 417–435. https://doi.org/10.1109/TPAMI.2011.142

Gathercole, C., & Ross, P. (1997). Small Populations over Many Generations can beat Large Populations over Few Generations in Genetic Programming. . In: Koza, J. R., et al., Eds.), Genetic Programming 1997: *Proceedings of the Second Annual Conference.*, 111-118. Stanford University, CA: Morgan Kaufmann,.

Gen, Mitsuo; Cheng, R. (2000). Genetic Algorithms & Engineering Optimization. *Wiley-Interscience Publications, New York, Estados Unidos*, 97–106.

Giordana.A., & Filippo. N. (1996). Search-intensive concept induction. *Evolutionary Computation,* (3(4)), 375-416. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.54.8986&rep=rep1&type=pdf

Goldberg. D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley, Reading, MA.

González A. Pérez R. (2001). Selection of relevant features in a fuzzy genetic learning algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B 31*, 417–425.

González J., Rojas I., Pomares H., Herrera L., Guillén A., P. J. R. F. (2007). Improving the accuracy while preserving the interpretability of fuzzy function approximators by means of multi-objective evolutionary algorithms. *International Journal of Approximate Reasoning*, *44(1):*, 32–44.

Gonzalez, P. (2007). Aprendizaje Evolutivo de Reglas Difusas para descripción de Subgrupos. Tesis Doctoral.Universidad de Granada.España.

Guerra Diego, A. (2015). *Estado del arte y análisis de métodos de optimización de recursos en plantas de producción.* Retrieved from https://uvadoc.uva.es/bitstream/10324/13230/1/TFG-P-272.pdf

Gunal S. y Edizkan R. (2008). Subspace based feature selection for pattern recognition. *Information Sciences*, *178(19):*, 3716–3726.

Gupta, M. (2012). Application of Weighted Particle Swarm Optimization in Association Rule Mining. *International Journal of Computer Sc Ience and Informatic*, *1*(3), 69–74.

Hand, D. J. (1981). *Driscrimination and Classification.* (Wiley., Ed.). Chichester, U.K.

Hasperue, W. (2013). Extraccion Del Conocimiento En Grandes Bases De Datos Utilizando Estrategias Adaptativas. In *Journal of Chemical Information and Modeling* (Vol. 53). https://doi.org/10.1017/CBO9781107415324.004

Hastie, T., Tibshirani, R., & Friedman, J. (2009). Machine Learning. In *Elements* (Vol. 27). https://doi.org/10.1007/978-0-387-84858-7

Hernandez, J., Ramirez, M., & Ferri, C. (2004). *Introducción a la Mineria de Datos* (D. F. Aragon, Ed.). madris: Hall, Pearson Prentice.

Hernández Riaño, V. L., López Pereira, J. M., & Hernández Riaño, H. E. (2013a). Minimización del retraso extremo a extremo en redes Unicast utilizando técnicas metaheurísticas. *Entre Ciencia e Ingeniería*, (14), 66–71. Retrieved from http://biblioteca.ucp.edu.co/ojs/index.php/entreciencia/article/view/2218

Hernández Riaño, V. L., López Pereira, J. M., & Hernández Riaño, H. E. (2013b). Minimización del retraso extremo

a extremo en redes Unicast utilizando técnicas metaheurísticas. *Entre Ciencia e Ingeniería*, (14), 66–71.

Herrera F., Lozano, M., & L. Verdegay J. (1998). Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. Artificial. *Intelligence Review 12: 265–319.*

Hincapie, R., Rios, C., & Gallego, R. (2004). Técnicas Heurísticas aplicadas al problema del cartero viajante (TSP). *SCIENCIA ET TECHNICA, 24*.

Ho S. Y., Chen H. M., H. S. J. y C. T. K. (2004). (2004). Design of accurate classifiers with a compact fuzzy-rule base using an evolu- tionary scatter partition of feature space. *IEEE Transactions on Systems, Man, and Cybernetics, Part B 34(*, 1031–1044.

Holland . H. J. (1975). Adaptation in natural and artificial systems. Ann arbor: *The University of Michigan Press.*

Holland, J. H. (1975). *Adaption in natural and artifical systems*. (53), 1975.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (2002). *A niched Pareto genetic algorithm for multiobjective optimization*. *1*, 82–87. https://doi.org/10.1109/icec.1994.350037

Hu Q., Yu D., L. J. y W. C. (2008). Neighborhood rough set based heterogeneous feature subset selection. *Information Sciences, 178(18):*, 3577–3594.

Iredi, S., Merkle, D., & Middendorf, M. (2001). Bi-Criterion Optimization with Multi Colony Ant Algorithms. *Lecture Notes in Computer Science 1993*, 359–372. https://doi.org/10.1007/3-540-44719-9_25

Ishibuchi. H., Nozaki. K., & Tanaka. H. (1994). Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms. *Fuzzy Sets and Systems, 65:*, 237–253.

Ishibuchi. H, Nakashima. T, & Nii. M. (2004). *Classification And Modeling With Linguistic Information Granules: Advanced Ap proaches To Linguistic Data Mining.* Springer Verlag.

Ishibuchi. H, Nozaki. K, & Tanaka. H. (1992). *Distributed representation of fuzzy rules and its application to pattern classification. Fuzzy Sets and Systems. 52*, 21–32.

Ishibuchi .H., Yamamoto. T., & Nakashima.T. (2005). Hybridization of fuzzy gbml approaches for pattern classification problems. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cy*, 359–365.

Ishibuchi H., Nozaki K., Y. N. y T. H. (1995). Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Transactions on Fuzzy Systems, 3(3):*, 260–270.

Ishibuchi H. y Nojima Y. (2007). Analysis of interpretability- accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics- based machine learning. *International Journal of Approximate Rea- Soning, 44(1):*, 4–31.

Ishibuchi H. y Yamamoto T. (2004). Fuzzy rule selection by multi- objective genetic local search algorithms and rule evaluation mea- sures in data mining. *Fuzzy Sets and Systems*, (141(1):), 59–88.

J., M. G. (1992). *Discriminant analysis and statistical patern recognition* (W. Interscience, Ed.).

J. Rua, Z. R. (2014). *ANÁLISIS DE MODELOS DE REDES NEURONALES ARTIFICIALES, PARA UN SISTEMA DE DIAGNÓSTICOS DE MIGRAÑAS CON AURA Y SIN AURA.* Barranquilla – Colombia.

J.R, Quinlan. (1993). *C4.5: PROGRAMS FOR MACHINE LEARNING J.* San Mateo, California.

Janikow, C.Z. (1993). A knowledge-intensive genetic algorithm for supervised learning. *Machine Learning, 13*, 189-228.

Kalsoom, A., Maqsood, M., Ghazanfar, M. A., Aadil, F., & Rho, S. (2018). A dimensionality reduction-based efficient software fault prediction using Fisher linear discriminant analysis (FLDA). In *Journal of Supercomputing* (Vol. 74). https://doi.org/10.1007/s11227-018-2326-5

Kangshun. Li, Yuanxiang. Li, Mo, H., & Chen, Z. (2005). A New Algorithm of Evolving Artificial Neural Networks via Gene Expression Programming. *Journal of the Korean Society for Industrial and Applied Mathematics*,

9(2), 83–90.

Kearns.M., & Mansour. Y. (1995). On the Boosting Ability of Top-Down Decision Tree Learning Algorithms. *In Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, 459–468.

Kennedy.Y:S:J:, & Eberhart. R. (2001). *Swarrm Intelligence* (Morgan. Kaufmann, Ed.).

Kim, M., Hiroyasu, T., Miki, M., & Watanabe, S. (2010). *SPEA2+: Improving the Performance of the Strength Pareto Evolutionary Algorithm 2*. 742–751. https://doi.org/10.1007/978-3-540-30217-9_75

Kohavi R. y John G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97(1-2):*, 273–324.

Konijn, R. M., Duivesteijn, W., Kowalczyk, W., & Knobbe, A. (2013). Discovering local subgroups, with an application to fraud detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 7818 LNAI*(PART 1), 1–12. https://doi.org/10.1007/978-3-642-37453-1_1

Koza J. R. (1992). *Genetic programming as a means for program- ming computers by natural selection. Statistics and Computing*. *4(2)*, 87–1.

Koza J. R. (1994). Genetic Programming II: Automatic Discovery of Reusable Programs. *The MIT Press, Cam-Bridge MA, USA*.

Kumar, P. G., Kavitha, M. S., & Ahn, B. C. (2016). Automated detection of cancer associated genes using a combined fuzzy-rough-set-based f-information and water swirl algorithm of human gene expression data. *PLoS ONE*, *11*(12), 1–24. https://doi.org/10.1371/journal.pone.0167504

Li, H., & Landa-Silva, D. (2008). Evolutionary multi-objective simulated annealing with adaptive and competitive search direction. *2008 IEEE Congress on Evolutionary Computation, CEC 2008*, 3311–3318. https://doi.org/10.1109/CEC.2008.4631246

Li R. P., M. M. y B. I. (2002). A fuzzy neural net- work for pattern classification and feature selection. *Fuzzy Sets and Systems, 130(1):*, 101–108.

Linfati, R., Escobar, J. W., & Gatica, G. (2014). Un algoritmo metaheurístico para el problema de localización y ruteo con flota heterogénea. *Ingeniería y Ciencia*, *10*(19), 55–76. https://doi.org/10.17230/ingciencia.10.19.3

Liu H. y Yu L. (2005). Toward integrating feature selection al- gorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering, 17(3):*, 491–502.

Lobo, R. S., & Castro, A. M. (2011). Cromatico, un nuevo metodo de optimizacion que se fundamenta a través de un algoritmo de búsqueda basada en la escala cromática de las notas músicales. *Universidad de Córdoba*.

Lu, Q., & Qiao, X. (2018). Sparse Fisher's linear discriminant analysis for partially labeled data. *Statistical Analysis and Data Mining*, *11*(1), 17–31. https://doi.org/10.1002/sam.11367

Luc, M., Marc, M., Ali, M., & Hammal, H. (2019). *Rank correlated subgroup discovery ´*.

Ma, J., Guo, D., Liu, M., Ma, Y., & Chen, S. (2009). Rules extraction from ANN based on clustering. *Proceedings of the 2009 International Conference on Computational Intelligence and Natural Computing, CINC 2009*, (2), 19–21. https://doi.org/10.1109/CINC.2009.168

Mahdianpari, M., Salehi, B., Mohammadimanesh, F., Brisco, B., Mahdavi, S., Amani, M., & Granger, J. E. (2018). Fisher Linear Discriminant Analysis of coherency matrix for wetland classification using PolSAR imagery. *Remote Sensing of Environment*, *206*(January 2017), 300–317. https://doi.org/10.1016/j.rse.2017.11.005

Martí, R. (2003). Procedimientos Metahurísticos en Optimización Combinatoria. *Valencia, Universidad de Valencia*.

Martinez. R. (2014). Metodologías Basadas en Minería de Datos para el Diseño y Optimización de Técnicas de Clasificación Automática. Universidad de Murcia.

Melián, B., Moreno Pérez, J. A., Marcos, J., & Vega, M. (2003). Metaheuristics: A global view Metaheurísticas: una

visión global. *Inteligencia Artificial Revista Iberoamericana de Inteligencia Artificial.*, *19*, 7–28. Retrieved from http://www.redalyc.org/pdf/925/92571901.pdf

Mendes, R.R.F., Voznika, F.B., Freitas, A.A., and Nievola, J. C. (2001). Discovering fuzzy classification rules with genetic programming and co-evolution,. *In Genetic and Evolutionary Computation Conference (GECCO-2001).*, 183-194.

Mendoza. M. (2016). *ESCAPE STRATEGIES ALGORITHM (ESSA) UN NUEVO ALGORITMO META HEURÍSTICO DE OPTIMIZACIÓN GLOBAL PARA PROBLEMAS DE VARIABLE REAL, INSPIRADO EN LA INTERACCIÓN DEPREDADOR-PRESA*. Universidad de Cordoba.

Merrikh-Bayat, F. (2015). The runner-root algorithm: A metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature. *Applied Soft Computing Journal*, *33*, 292–303. https://doi.org/10.1016/j.asoc.2015.04.048

Michalewicz Z. (1996). Genetic algorithms + data structures = evo- lution programs. *Springer-Verlag*.

Mikut. R., Jäkel..J., & Gröll. L. (2005). Interpretability issues in data- based learning of fuzzy systems. *Fuzzy Sets and Systems*, *150(2):*, 179–197.

Monteserin, A., & Armentano, M. G. (2018). Influence-based approach to market basket analysis. *Information Systems*, *78*, 214–224. https://doi.org/10.1016/j.is.2018.01.008

Montgomery, D., Jennings, C., & Kulahci, M. (2008). *Introduction to time series analysis and foecasting*.

Morales. E, & Mariano. E. (1999). A Multiple Objective Ant-Q Algorithm for the Design of Water Distribution Irrigation Networks. *Instituto Mexicano de Tecnolog__a Del Agua*, (November 1999).

Mori, Y., & Suzuki, T. (2018). Generalized ridge estimator and model selection criteria in multivariate linear regression. *Journal of Multivariate Analysis*, *165*, 243–261. https://doi.org/10.1016/j.jmva.2017.12.006

Morillo Torres, D., Moreno, L., & Díaz, J. (2014). Metodologías Analíticas y Heurísticas para la Solución del Problema de Programaciónde Tareas con Recursos Restringidos (RCPSP): una revisión. Parte 2. *Ingeniería y Ciencia*, *10*(20), 203–227. https://doi.org/10.17230/ingciencia.10.20.12

Motoda, H., Wu, X., Kumar, V., McLachlan, G. J., Zhou, Z.-H., Hand, D. J., … Ng, A. (2007). Top 10 algorithms in data mining. In *Knowledge and Information Systems* (Vol. 14). https://doi.org/10.1007/s10115-007-0114-2

Naghash Asadi, A., Abdollahi Azgomi, M., & Entezari-Maleki, R. (2019). Unified power and performance analysis of cloud computing infrastructure using stochastic reward nets. *Computer Communications*, *138*(June 2018), 67–80. https://doi.org/10.1016/j.comcom.2019.03.004

Nauck D. Kruse R. (1997). A neuro-fuzzy method to learn fuzzy classification rules from data. *Fuzzy Sets and Systems*, *89:*, 277–288.

Naudin, M., Tremblais, B., Guillevin, C., Guillevin, R., & Fernandez, C. (2002). Diffuse Low Grade Glioma NMR Assessment for Better Intra-operative Targeting Using Fuzzy Logic. *Computers & Graphics*, *26*(3), 528. https://doi.org/10.1016/s0097-8493(02)00077-8

Nieto, H. (2011). *DISEÑO E IMPLEMENTACION DE UNA METAHEURISTICA HIBRIDA BASADA EN RECOCIDO SIMULADO, ALGORITMOS GENETICOS Y TEORIA DE AUTOMATAS PARA LA OPTIMIZACION BI-OBJETIVO DE PROBLEMAS COMBINATORIOS*. Universidad del Norte, Barranquilla, Colombia.

Niño. (2012). Evolutionary Algorithms based on the Automata Theory for the Multi-objective Optimization of Combinatorial Problems. *INT J COMPUT COMMUN, 7 (5)*(1), 916–923. Retrieved from http://journal.univagora.ro/download/pdf/644.pdf

Niño., E., & Ardila., C. (2009). Algoritmo basado basado en la obtencion de optimos globales en problemas combnatorios. *Ingenieria y Desarrollo*, *25*, 1–11.

Niño, E. (2010). *Diseño e implementacion de una metaheuristica badasa en automatas finitos determistas para la*

optimizacion multiobjetivo de problemas Combinatorios. Universidad del Norte, Barranquilla, Colombia.

Niño, E. (2012). SAMODS and SAGAMODS: Novel Algorithms Based on the Automata Theory for the Multiobjective Optimization of Combinatorial Problems. *International Journal of Artificial Intelligence*, *8 S12*. Retrieved from http://ceser.in/ceserp/index.php/ijai/article/view/1287

Niño, E., Ardila, C., Molinares, D., Barrios, A., & Yesid, D. (2011). MODS: A Novel Metaheuristic of Deterministic Swapping for the Multi-Objective Optimization of Combinatorials Problems. *Computer Tecnology and Application*, *2*, 280–292.

Niño, E., Nieto, H., & Chinchilla, A. (2011). EMSA : Hybrid Metaheuristic based on Genetic Algorithms , Simulated Annealing and Deterministic Swapping. *Universidad Del Norte*, (4), 1–12.

Noda, E., Freitas, A.A., and Lopes, H. . (1999). Discovering Interesting Prediction Rules with a Genetic Algorithm. *In IEEE Congress on Evolutionary Computation (CEC 99). Washington, USA*.

Noda, E.; Freitas, A.A., and Lopes, H. S. (1999). Discovery Interesting Prediction Rules with a Genetic Algorithm. *In IEEE Congress on Evolutionary Computation (CEC99).*, Washington. USA.

Oberlin.P, Rathinam.S, & Darbha.S. (2009). A transformation for a Multiple Depot, Multiple Traveling Salesman Problem. *American Control Conference*, 2636–2641.

Olivas, F., Valdez, F., Melin, P., Sombra, A., & Castillo, O. (2019). Interval type-2 fuzzy logic for dynamic parameter adaptation in a modified gravitational search algorithm. *Information Sciences*, *476*, 159–175. https://doi.org/10.1016/j.ins.2018.10.025

Papadakis S. E. y Theocharis J. B. (2006). A genetic method for designing TSK models based on objective weighting: application to classification problems. *Soft Computing, 10(9):*, 805–824.

Parthasarathy, S., Zaki, M. J., Ogihara, M., & Li, W. (1997). *New Algorithms for Fast Discovery of Association*. 283–286.

Pearl J. (1988). Probabilistic reasoning in intelligent systems. Mor- gan Kaufmann, Palo Alto, USA. In *Mor- gan Kaufmann, Palo Alto, USA.*

Pérez-Ortega, J., Castillo-Zacatelco, H., Vilariño-Ayala, D., Mexicano-Santoyo, A., Zavala-Díaz, J. C., Martínez-Rebollar, A., & Estrada-Esquivel, H. (2016). Una nueva estrategia heurística para el problema de Bin Packing. *Ingeniería, Investigación y Tecnología*, *17*(2), 155–168. https://doi.org/10.1016/j.riit.2016.06.001

Pérez. E, Herrera. F., & Hernández. C. (2003). Finding multiple solu- tions in job shop scheduling by niching genetic algorithms. *Journal of Intelligent Manufacturing*, *14(3-4):*, 323–339.

Pérez. M, Pérez, . P, Rivera. A, Jesus. M, & López. P. (2010). CO 2 RBFN : predicción de series temporales con un enfoque cooperativo-competitivo. *Soft Computing 14*, *14*, 953–971. Retrieved from https://sci2s.ugr.es/keel/pdf/keel/congreso/maeb09.pdf

Pham, B. T., & Prakash, I. (2019). Evaluation and comparison of LogitBoost Ensemble, Fisher's Linear Discriminant Analysis, logistic regression and support vector machines methods for landslide susceptibility mapping. *Geocarto International*, *34*(3), 316–333. https://doi.org/10.1080/10106049.2017.1404141

Provost, F., & Fawcett, T. (2001). Robust classification for imprecise environments. *Machine Learning*, *42*(3), 203–231. https://doi.org/10.1023/A:1007601015854

Quinlan. J. R. (1986). Induction of Decision Trees. *In Machine Learning, 1*, 81-106.

Quinlan. J. R. (1993). C4.5: Programs for Machine Learning. In *Morgan Kaufmann Publishers*.

Ramírez. J, Osuna. I, Rojas. J, & Guerrero. S. (2016). Remuestreo Bootstrap y Jackknife en confiabilidad: Caso Exponencial y Weibull. *Revista Facultad De Ingeniería*, *25*(41), 55. https://doi.org/10.19053/01211129.4137

Ranjan, R., & Sharma, A. (2019). Evaluation of Frequent Itemset Mining Platforms using Apriori and FP- Growth Algorithm. *Department of Computer Science and Engineering, DIT University*, (February), 1–6.

Ravi V., R. P. J. y Z. H. J. (2000). Pattern clas- sification with principal component analysis and fuzzy rule bases. *European Journal of Operational Research, 126(3):*, 526–533.

Ravi V. y Zimmermann H. J. (2001). A neural network and fuzzy rule base hybrid for pattern classification. *Soft Computing, 5(2):*, 152–159.

Riaño. J., Prieto. F., Sánchez. E, C, A., & Castellano. G. (2010). Analysis and convergence of weighted dimensionality reduction methods. *Revista Facultad De Ingeniería. Universidad de Antioquia*, 1–11.

Rocha, L.; González, C. y Orjuela, J. (2011). Una revisión al estado del arte del problema de ruteo de vehículos: Evolución histórica y métodos de solución. *Ingeniería, Vol. 16, N*, 35-55 .

Rodríguez.C, Pena. M, & Piñero. P. (2015). Aprendizaje de reglas difusas usando algoritmos genéticos. *Congreso Internacional de Matemática y Computación COMPUMAT, At La Habana, Cuba*.

Romao, W., Freitas, A.A., and Pacheco, R. C. S. (2002a). A Genetic Algorithm for Discovering Interesting Fuzzy Prediction Rules: applications to science and technology data. *In Genetic and Evolutionary Computation Conference (GECCO*.

Romao, W., Freitas, A.A., and Pacheco, R. C. S. (2002b). A Genetic Algorithm for Discovering Interesting Fuzzy Prediction Rules: applications to science and technology data. *In Genetic and Evolutionary Computation Conference (GECCO 2002).*

Romero, G. P., & Niz, A. P. (2013). facultad Ingeniería de Sistemas , con técnicas de minería de datos. *Https://Revistas.Unisimon.Edu.Co*, 1–6. Retrieved from https://revistas.unisimon.edu.co/index.php/identic/article/view/2484

Roubos J. A., S. M. y A. J. (2003). Learning fuzzy clas- sification rules from labeled data. *Information Sciences, 150(1-2):*, 77–93.

Ruiz, J. (2013). *ENTRENAMIENTO DE REDES NEURONALES ARTIFICIALES BASADO EN ALGORITMO EVOLUTIVO Y TEORÍA DE AUTÓMATAS FINITOS*. tesis de maestia sin publicacion. Universidad del Norte, Barranquilla, Colombia.

S. Garcia., J .Derrac., F. Herrera., & D. Molina. (2012). *Un tutorial sobre el uso de test estadısticos no parametricos en comparaciones multiples de metaheurısticas y algoritmos evolutivos*. 8–10.

Sampieri, R. H., Collado, C. F., & Baptista, M. del pilar. (2010). *METODOLOGIA DE LA INVESTIGACIÓN* (QUINTA EDI). Mc Graw Hill.

Schwefel H. P. (1995). Evolution and Optimum Seeding. *Wiley Inc*.

Setnes M. Roubos H. (2000). GA-fuzzy modeling and classifica- tion: complexity and performance. *IEEE Transactions on Fuzzy Systems, 8(5):*, 509–522.

Setnes M. y Babuška R. (2001). Rule base reduction: Some com- ments on the use of orthogonal transforms. *IEEE Transactions on Systems, Man, and Cybernetics, Part B 31:*, 199–206.

Seymour, L., Brockwell, P. J., & Davis, R. A. (2006). Introduction to Time Series and Forecasting. In *Journal of the American Statistical Association* (Vol. 92). https://doi.org/10.2307/2965440

Shen Q. Jensen R. (2004). Selecting informative features with fuzzy-rough sets and its application for complex systems monitor ing. *Pattern Recognition, 37(7):*, 1351–1363.

Sheskin, D. J. (2000). *PARAMETRIC and NONPARAMETRIC STATISTICAL PROCEDURES SECOND EDITION*. Retrieved from www.crcpress.com

Silipo R. y Berthold M. R. (2000). Input features' impact on fuzzy decision processes. *IEEE Transactions on Systems, Man, and Cy- Bernetics, Part B 30(*, 821–834.

Simon, C., Weber, P., & Evsukoff, A. (2008). Bayesian networks inference algorithm to implement Dempster Shafer theory in reliability analysis. *Reliability Engineering and System Safety*, *93*(7), 950–963.

https://doi.org/10.1016/j.ress.2007.03.012

Street, W. (1982). SOME PROBLEMS WITH FROM FINITE MIXTURE DISTRIBUTIONS. *Contract*.

Sui. X, & Leung. H. (2008). An Adaptive Bidding Strategy in Multi-round Combinatorial Auctions for Resource Allocation," Tools with Artificial Intelligence. *ICTAI '08. 20th IEEE International Conference*, *2*, 423–430. https://doi.org/978-0-7695-3440-4

Tovar, L., Coronell, M., & Donoso, Y. (2007). Optimización multiobjetivo en redes ópticas con transmisión Multicast, utilizando algoritmos evolutivos y lógica difusa. *Ingeniería & Desarrollo*, *21*.

Tsakiridis, N. L., Theocharis, J. B., & Zalidis, G. C. (2016). DECO3R: A Differential Evolution-based algorithm for generating compact Fuzzy Rule-based Classification Systems. *Knowledge-Based Systems*, *105*, 160–174. https://doi.org/10.1016/j.knosys.2016.05.013

Ulungu, E. L., Teghem, J., Fortemps, P. H., & Tuyttens, D. (1999). MOSA method: A tool for solving multiobjective combinatorial optimization problems. *Journal of Multi-Criteria Decision Analysis*, *8*(4), 221–236. https://doi.org/10.1002/(SICI)1099-1360(199907)8:4<221::AID-MCDA247>3.0.CO;2-O

Vaezpour, E., Dehghan, M., & Yousefi'zadeh, H. (2019). Robust joint user association and resource partitioning in heterogeneous cloud RANs with dual connectivity. *Computer Communications*, *138*, 1–10. https://doi.org/10.1016/j.comcom.2019.02.008

Van Broekhoven E., A. V. y D. B. B. (2007). Interpretability-preserving genetic optimization of linguistic terms in fuzzy models for fuzzy ordered classification: An ecological case study. *International Journal of Approximate Reasoning*, *44(1):*, 65– 90.

Vidya, V., & Nedunchezhian, R. (2011). A robust weighted association rule mining using FP-tree. *European Journal of Scientific Research*, *66*(4), 600–609.

Wang J.-S. Lee C. S. G. (2000). Self-adaptive neuro-fuzzy infer- ence systems for classification applications. *IEEE Transactions on Fuzzy Systems*, *10(6):*, 790–802.

Weise, T. (2008). *Global Optimization Algorithms Theory and Application* (segunda ed).

Weiss, S. I., & Kulikowski, C. (1991). *Computer System Than Learn: Classification y Prediction Methods from Statistic, Neural Networks, Machine Learning, and Expert System*. San francisco, California: Morgan Kaufmann.

Whitley L. D. y Kauth J. (1988). GENITOR: A different genetic algorithm. *En Proceedings of the Rocky Mountain Conference on Artificial Intelligence*, páginas 118–130. Denver, USA.

Witten, I. H., & Frank, E. (2000). *Machine Learning Algorithms in Java Nuts and bolts*. 58. Retrieved from https://www.cs.ru.nl/P.Lucas/teaching/DM/weka.pdf

Y. F. Cabrera. (2011). *MÉTODOS DE APRENDIZAJE PARA DOMINIOS CON DATOS MEZCLADOS BASADOS EN LA TEORÍA DE LOS CONJUNTOS APROXIMADOS EXTENDIDA*. Universidad Central Marta Abreu.

Yañez. E. (2009). *T ecnicas de Auto-Adaptacion para Algoritmos Evolutivos Multi-objetivo*. CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITECNICO NACIONAL DEPARTAMENTO DE COMPUTACION.

Yen J. Wang L. (1999). Simplifying fuzzy rule-based models us- ing orthogonal transformation methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part B 29:*, 13–24.

Yu S., B. S. D. y S. P. (2002). Genetic feature selection combined with composite fuzzy nearest neighbor classifiers for hy- perspectral satellite imagery. *Pattern Recognition Letters*, *23(1-3):*, 183–190.

Zadeh L. A. (1975). The concept of a linguistic variable and its applications to approximate reasoning, parts I, II, III. *Information Sciences*, 199–249, 301–357, 43–80.

Zadeh, L.A. (1965). fuzzy sets. *Journal of Plant Pathology*, 338–353.

Zadeh, L.A. (1973). Outline of a New Approach to the Analysis of Complex Systems and Decision Processes. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-3,* 28-44. https://doi.org/https://doi.org/10.1109/TSMC.1973.5408575

Zadeh, Lotfi A. (1994). Soft Computing and Fuzzy Logic. *IEEE Software*, *11*(6), 48–56. https://doi.org/10.1109/52.329401

Zhang, J., Wang, Y., & Feng, J. (2013). Attribute index and uniform design based multiobjective association rule mining with evolutionary algorithm. *The Scientific World Journal*, *2013*. https://doi.org/10.1155/2013/259347

Zhang, X., Wang, C., & Wu, Y. (2018). Functional envelope for model-free sufficient dimension reduction. *Journal of Multivariate Analysis*, *163*, 37–50. https://doi.org/10.1016/j.jmva.2017.09.010

Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C.: da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation, Vol 7*, *No*, pag. 117–132,.

9. **Anexos 1**

Test no paramétricos para el análisis estadístico de los resultados.

En esta sección se explica el test estadístico no paramétrico que se ha utilizado en este trabajo para el análisis de los resultados obtenidos por el método en estudio.

Test de Friedman

El test de Friedman (Sheskin, 2000), es un test no paramétrico de medidas repetidas, opera de la siguiente forma: El primer paso consiste en convertir los resultados obtenidos por cada algoritmo a rangos. Para ello se ordenan los k algoritmos separadamente para cada problema según el rendimiento alcanzado, asignando rangos de forma ascendente, es decir, 1 al mejor algoritmo, 2 al segundo mejor, y así sucesivamente. En el caso de que varios algoritmos obtengan el mismo rendimiento para esa instancia se asignaran rangos medios. Así, sea r_j^i el rango del j-ésimo algoritmo sobre i-ésimo problema.

El test de Friedman compara los rangos medios de los algoritmos sobre *n* problemas, el cálculo es el siguiente:

$$\boldsymbol{R_j = \frac{1}{n}\sum_{i=1}^{n} r_j^i}$$

Bajo la hipótesis nula, el rendimiento de todos los algoritmos es el mismo y los rangos R_j deben ser similares. El estadístico

$$X_F^2 = \frac{12n}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right]$$

El test de Friedman, se distribuye de acuerdo a una distribución X^2 con k-1 grados de libertad, cuando *n* y *k* son suficientemente grandes (como regla experimental práctica, $n > 10$ y $k > 5$). Para

un número de algoritmos y casos se han calculado los valores críticos exactos en (Derrac, Herrera, & García, 2010), (S. Garcia. et al., 2012).

10. **Apéndices**

Apéndice A. Descripción del conjunto de datos del repositorio Foodmart utilizado en la experimentación.

En la tabla A.14 se muestra las propiedades del conjunto de datos utilizada en la experimentación de esta investigación, la base de datos de prueba está disponible en: http://pentaho.dlpage.phiintegration.com/mondrian/mysql-foodmart-database.

Tabla A.134.Propiedades de la base de dato el repositorio

Investigadores	Número de transacciones	Descripción
Foodmart	251,357	Registros con número de factura. Fecha de compra Código de artículos Nombre de los artículos Cantidad Precio

A continuación, se describe brevemente el conjunto de datos:

Foodmart: Este conjunto de datos de compra de productos de supermercado contiene 51,357 registros de compra de los clientes en los años 1997 y 1998.con atributos numéricos y categóricos.

Apéndice B. Tablas de experimentación sobre el conjunto de datos del repositorio Foodmart.

Este apéndice contiene las tablas correspondientes a las experimentaciones realizadas para el algoritmo EKCEAD, reflejándose en ella las ejecuciones realizadas. En cada tabla se observan los resultados de la experimentación para obtener reglas DNF

Las 8 primeras columnas indican los atributos del antecedente.

Los 2 restantes columnas indican los atributos del consecuente.

Tabla A145. Resumen de la experimentación de EKCEAD inicio (0 ejecución)

27	215	38	1540	265	694	1315	672	458	836
820	1541	527	225	1194	1175	401	1001	742	1301
790	525	885	689	186	1195	1089	1086	1045	90
1496	1529	1293	1143	399	789	267	477	119	375
1039	383	848	1459	765	377	91	542	1319	1378
950	315	1420	879	638	173	11	972	1256	1200
1278	510	831	1294	827	1228	712	349	740	715
1363	1404	328	655	1056	1064	1551	179	1270	969
340	41	856	1012	13	947	1169	937	933	325
1144	1038	708	749	286	451	346	35	1224	502
501	668	992	206	1361	281	35	248	117	293
1412	1410	1021	678	186	1306	1188	682	56	769
328	1030	172	17	648	887	153	785	1210	65
368	1487	120	358	1212	60	1439	109	1103	322
1041	238	1401	1315	17	313	881	1038	822	1378
646	1237	1450	674	779	903	479	1166	1108	6
298	1168	674	1492	1181	1139	783	82	1500	1395
884	247	371	745	1522	279	174	767	1380	299
884	510	1375	1062	1034	568	894	1100	1132	358
57	197	1272	159	999	1124	124	1348	868	1226
985	1449	1366	1334	205	1276	739	292	1440	486
1066	727	1320	74	14	1244	49	358	612	1261

227	867	236	86	1390	961	111	422	862	335
1471	1528	374	19	1019	1234	449	215	293	1498
799	359	1545	291	398	1075	669	836	1154	1193
594	1191	267	213	833	736	210	272	1416	470
1507	516	268	1038	442	1460	1230	164	432	1465
1067	727	83	1034	1509	461	350	799	837	136
1472	1487	1414	1281	353	633	1200	87	587	1084
1517	1332	1189	661	918	1508	773	669	129	1103
1174	687	579	1147	153	683	527	130	660	1075
360	1357	1120	326	710	287	298	641	295	720
797	1175	600	48	523	13	163	1483	811	1223
775	1166	517	11	869	239	750	1067	835	616
832	540	827	194	788	105	462	414	893	270
1507	1199	677	593	1128	1501	239	543	1067	545
1331	567	816	893	680	269	945	851	1469	1543
966	870	1338	289	501	1109	989	1229	554	1085
372	368	1406	907	525	891	1418	1144	182	277
248	515	1430	819	1316	1122	1143	318	112	1054
258	1400	939	1305	1537	229	457	773	1081	397
1275	1341	186	329	227	21	1048	1091	600	678
1500	874	1284	812	1430	470	527	1434	1273	171
1217	164	769	346	429	82	847	493	1277	715
1266	97	906	1162	1387	905	891	515	1016	1181
377	164	488	838	432	1110	1135	1359	803	686
1560	752	396	1154	796	116	1346	716	250	617
988	1496	299	683	1304	604	1419	62	659	357
1168	352	194	392	533	132	1281	157	602	224
375	362	282	209	381	521	48	173	800	683

Tabla A156. Resumen de la experimentación de EKCEAD para 10 ejecuciones

27	215	38	1540	265	694	1315	672	458	836
820	1541	527	225	1194	1175	401	1001	742	1301
790	525	885	689	186	1195	1089	1086	1045	90
1496	1529	1293	1143	399	789	267	477	119	375
1039	383	848	1459	765	377	91	542	1319	1378
950	315	1420	879	638	173	11	972	1256	1200
1278	510	831	1294	827	1228	712	349	740	715
1363	1404	328	655	1056	1064	1551	179	1270	969
868	57	197	1378	1272	159	848	124	1226	1124
1144	1038	708	749	286	451	346	35	1224	502
501	668	992	206	1361	281	35	248	117	293
1412	1410	1021	678	186	1306	1188	682	56	769
328	1030	172	17	648	887	153	785	1210	65
368	1487	120	358	1212	60	1439	109	1103	322
600	1483	523	48	163	298	1492	1395	13	1223
646	1237	1450	674	779	903	479	1166	1108	6
298	1168	674	1492	1181	1139	783	82	1500	1395
884	247	371	745	1522	279	174	767	1380	299
884	510	1375	1062	1034	568	894	1100	1132	358
57	197	1272	159	999	1124	124	1348	868	1226
985	1449	1366	1334	205	1276	739	292	1440	486
1175	1195	600	1086	1045	90	1089	790	797	163
163	1305	13	457	523	229	1081	939	797	811
1471	1528	374	19	1019	1234	449	215	293	1498
799	359	1545	291	398	1075	669	836	1154	1193
594	1191	267	213	833	736	210	272	1416	470
1507	516	268	1038	442	1460	1230	164	432	1465
1067	727	83	1034	1509	461	350	799	837	136
1472	1487	1414	1281	353	633	1200	87	587	1084
1517	1332	1189	661	918	1508	773	669	129	1103
1174	687	579	1147	153	683	527	130	660	1075
360	1357	1120	326	710	287	298	641	295	720
797	1175	600	48	523	13	163	1483	811	1223
281	248	1168	82	293	1395	206	1181	117	35
832	540	827	194	788	105	462	414	893	270
1507	1199	677	593	1128	1501	239	543	1067	545
1331	567	816	893	680	269	945	851	1469	1543
966	870	1338	289	501	1109	989	1229	554	1085
372	368	1406	907	525	891	1418	1144	182	277
248	515	1430	819	1316	1122	1143	318	112	1054

258	1400	939	1305	1537	229	457	773	1081	397
1275	1341	186	329	227	21	1048	1091	600	678
1500	874	1284	812	1430	470	527	1434	1273	171
1217	164	769	346	429	82	847	493	1277	715
1266	97	906	1162	1387	905	891	515	1016	1181
501	248	1341	992	117	600	1361	281	1091	35
1560	752	396	1154	796	116	1346	716	250	617
1122	831	112	712	1430	1278	1143	1228	510	515
1168	352	194	392	533	132	1281	157	602	224
1496	1529	1293	412	399	789	267	477	119	375

Tabla A167. Resumen de la experimentación de EKCEAD para 20 ejecuciones

27	215	38	1540	265	694	1315	672	458	836
493	346	1277	769	1434	812	1430	1500	874	82
790	525	885	689	186	1195	1089	1086	1045	90
1496	1529	1293	1143	399	789	267	477	119	375
1039	383	848	1459	765	377	91	542	1319	1378
950	315	1420	879	638	173	11	972	1256	1200
1278	510	831	1294	827	1228	712	349	740	715
1363	1404	328	655	1056	1064	1551	179	1270	969
868	57	197	1378	1272	159	848	124	1226	1124
1144	1038	708	749	286	451	346	35	1224	502
501	668	992	206	1361	281	35	248	117	293
1412	1410	1021	678	186	1306	1188	682	56	769
328	1030	172	17	648	887	153	785	1210	65
368	1487	120	358	1212	60	1439	109	1103	322
600	1483	523	48	163	298	1492	1395	13	1223
646	1237	1450	674	779	903	479	1166	1108	6
298	1168	674	1492	1181	1139	783	82	1500	1395
884	247	371	745	1522	279	174	767	1380	299
884	510	1375	1062	1034	568	894	1100	1132	358
57	197	1272	159	999	1124	124	1348	868	1226
985	1449	1366	1334	205	1276	739	292	1440	486
1175	1195	600	1086	1045	90	1089	790	797	163
163	1305	13	457	523	229	1081	939	797	811
1471	1528	374	19	1019	1234	449	215	293	1498
799	359	1545	291	398	1075	669	836	1154	1193
1067	1081	727	773	457	1034	1305	837	799	461
1507	516	268	1038	442	1460	1230	164	432	1465
1067	727	83	1034	1509	461	350	799	837	136
1472	1487	1414	1281	353	633	1200	87	587	1084

668	1193	117	35	799	992	291	1075	206	398
1174	687	579	1147	153	683	527	130	660	1075
847	164	82	90	600	346	493	715	1195	1045
797	1175	600	48	523	13	163	1483	811	1223
281	248	1168	82	293	1395	206	1181	117	35
832	540	827	194	788	105	462	414	893	270
516	248	1465	1038	432	1181	1230	1507	164	293
1331	567	816	893	680	269	945	851	1469	1543
966	870	1338	289	501	1109	989	1229	554	1085
372	368	1406	907	525	891	1418	1144	182	277
248	515	1430	819	1316	1122	1143	318	112	1054
258	1400	939	1305	1537	229	457	773	1081	397
1275	1341	186	329	227	21	1048	1091	600	678
1500	874	1284	812	1430	470	527	1434	1273	171
1217	164	769	346	429	82	847	493	1277	715
1266	97	906	1162	1387	905	891	515	1016	1181
501	248	1341	992	117	600	1361	281	1091	35
1450	1212	368	1166	479	109	1237	646	779	6
1122	831	112	712	1430	1278	1143	1228	510	515
1168	352	194	392	533	132	1281	157	602	224
797	906	1016	457	1162	97	13	523	1387	905

Tabla A178. Resumen de la experimentación de EKCEAD para 30 ejecuciones

27	215	38	1540	265	694	1315	672	458	836
493	346	1277	769	1434	812	1430	1500	874	82
790	525	885	689	186	1195	1089	1086	1045	90
1496	1529	1293	1143	399	789	267	477	119	375
848	377	383	1378	1162	1459	97	906	891	1266
1266	97	1306	377	1162	906	56	1021	678	682
1278	510	831	1294	827	1228	712	349	740	715
461	90	600	773	457	163	1195	1175	1089	1067
868	57	197	1378	1272	159	848	124	1226	1124
1144	1038	708	749	286	451	346	35	1224	502
501	668	992	206	1361	281	35	248	117	293
1412	1410	1021	678	186	1306	1188	682	56	769
328	1030	172	17	648	887	153	785	1210	65
885	1195	525	1045	1229	90	1086	966	186	1089
600	1483	523	48	163	298	1492	1395	13	1223
646	1237	1450	674	779	903	479	1166	1108	6
298	1168	674	1492	1181	1139	783	82	1500	1395
884	247	371	745	1522	279	174	767	1380	299

884	510	1375	1062	1034	568	894	1100	1132	358
57	197	1272	159	999	1124	124	1348	868	1226
985	1449	1366	1334	205	1276	739	292	1440	486
1175	1195	600	1086	1045	90	1089	790	797	163
163	1305	13	457	523	229	1081	939	797	811
1471	1528	374	19	1019	1234	449	215	293	1498
799	359	1545	291	398	1075	669	836	1154	1193
1067	1081	727	773	457	1034	1305	837	799	461
1387	1276	1162	457	905	1449	523	906	797	13
1067	727	83	1034	1509	461	350	799	837	136
600	811	13	992	1175	248	281	523	163	1341
668	1193	117	35	799	992	291	1075	206	398
1174	687	579	1147	153	683	527	130	660	1075
847	164	82	90	600	346	493	715	1195	1045
797	1175	600	48	523	13	163	1483	811	1223
281	248	1168	82	293	1395	206	1181	117	35
832	540	827	194	788	105	462	414	893	270
1545	13	1175	1223	799	836	523	1483	163	398
1331	567	816	893	680	269	945	851	1469	1543
966	870	1338	289	501	1109	989	1229	554	1085
372	368	1406	907	525	891	1418	1144	182	277
945	186	1543	269	680	525	893	816	851	1331
258	1400	939	1305	1537	229	457	773	1081	397
1275	1341	186	329	227	21	1048	1091	600	678
1500	874	1284	812	1430	470	527	1434	1273	171
1217	164	769	346	429	82	847	493	1277	715
1266	97	906	1162	1387	905	891	515	1016	1181
501	248	1341	992	117	600	1361	281	1091	35
1450	1212	368	1166	479	109	1237	646	779	6
1122	831	112	712	1430	1278	1143	1228	510	515
1168	352	194	392	533	132	1281	157	602	224
797	906	1016	457	1162	97	13	523	1387	905

Tabla A189. Resumen de la experimentación de EKCEAD para 40 ejecuciones

27	215	38	1540	265	694	1315	672	458	836
493	346	1277	769	1434	812	1430	1500	874	82
790	525	885	689	186	1195	1089	1086	1045	90
1496	1529	1293	1143	399	789	267	477	119	375
848	377	383	1378	1162	1459	97	906	891	1266
1266	97	1306	377	1162	906	56	1021	678	682
1278	510	831	1294	827	1228	712	349	740	715
461	90	600	773	457	163	1195	1175	1089	1067
868	57	197	1378	1272	159	848	124	1226	1124
1144	1038	708	749	286	451	346	35	1224	502
501	668	992	206	1361	281	35	248	117	293
1412	1410	1021	678	186	1306	1188	682	56	769
328	1030	172	17	648	887	153	785	1210	65
885	1195	525	1045	1229	90	1086	966	186	1089
939	740	163	523	1294	712	1305	229	13	827
646	1237	1450	674	779	903	479	1166	1108	6
298	1168	674	1492	1181	1139	783	82	1500	1395
1162	905	97	1143	375	399	1293	119	1266	906
884	510	1375	1062	1034	568	894	1100	1132	358
885	790	1089	82	689	1086	206	1195	186	117
985	1449	1366	1334	205	1276	739	292	1440	486
1175	1195	600	1086	1045	90	1089	790	797	163
163	1305	13	457	523	229	1081	939	797	811
1471	1528	374	19	1019	1234	449	215	293	1498
799	359	1545	291	398	1075	669	836	1154	1193
1067	1081	727	773	457	1034	1305	837	799	461
1387	1276	1162	457	905	1449	523	906	797	13
1067	727	83	1034	1509	461	350	799	837	136
600	811	13	992	1175	248	281	523	163	1341
668	1193	117	35	799	992	291	1075	206	398
1174	687	579	1147	153	683	527	130	660	1075
847	164	82	90	600	346	493	715	1195	1045
797	1175	600	48	523	13	163	1483	811	1223
281	248	1168	82	293	1395	206	1181	117	35
479	906	1237	1016	457	523	903	905	1108	6
669	1193	769	1154	874	82	799	398	812	1075
1331	567	816	893	680	269	945	851	1469	1543
1166	486	739	292	903	1276	1334	1108	646	1449
372	368	1406	907	525	891	1418	1144	182	277
945	186	1543	269	680	525	893	816	851	1331
258	1400	939	1305	1537	229	457	773	1081	397
1275	1341	186	329	227	21	1048	1091	600	678
1500	874	1284	812	1430	470	527	1434	1273	171

1217	164	769	346	429	82	847	493	1277	715
1266	97	906	1162	1387	905	891	515	1016	1181
501	248	1341	992	117	600	1361	281	1091	35
1450	1212	368	1166	479	109	1237	646	779	6
523	163	13	1166	368	1341	281	479	811	1450
1168	352	194	392	533	132	1281	157	602	224
797	906	1016	457	1162	97	13	523	1387	905

Tabla A20. Resumen de la experimentación de EKCEAD para 50 ejecuciones

27	215	38	1540	265	694	1315	672	458	836
493	346	1277	769	1434	812	1430	1500	874	82
790	525	885	689	186	1195	1089	1086	1045	90
1496	1529	1293	1143	399	789	267	477	119	375
848	377	383	1378	1162	1459	97	906	891	1266
328	1030	172	17	648	887	153	785	1210	196
1278	510	831	1294	827	1228	712	349	740	715
461	90	600	773	457	163	1195	1175	1089	1067
868	57	197	1378	1272	159	848	124	1226	1124
1144	1038	708	749	286	451	346	35	1224	502
501	668	992	206	1361	281	35	248	117	293
1412	1410	1021	678	186	1306	1188	682	56	769
1543	893	851	372	680	277	525	269	1144	1418
885	1195	525	1045	1229	90	1086	966	186	1089
939	740	163	523	1294	712	1305	229	13	827
646	1237	1450	674	779	903	479	1166	1108	6
298	1168	674	1492	1181	1139	783	82	1500	1395
1162	905	97	1143	375	399	1293	119	1266	906
884	510	1375	1062	1034	568	894	1100	1132	358
885	790	1089	82	689	1086	206	1195	186	117
985	1449	1366	1334	205	1276	739	292	1440	486
1175	1195	600	1086	1045	90	1089	790	797	163
163	1305	13	457	523	229	1081	939	797	811
674	1195	1089	1168	90	186	298	1045	1500	783
799	359	1545	291	398	1075	669	836	1154	1193
1067	1081	727	773	457	1034	1305	837	799	461
1387	1276	1162	457	905	1449	523	906	797	13
1067	727	83	1034	1509	461	350	799	837	136
600	811	13	992	1175	248	281	523	163	1341
668	1193	117	35	799	992	291	1075	206	398
1174	687	579	1147	153	683	527	130	660	1075

847	164	82	90	600	346	493	715	1195	1045
797	1175	600	48	523	13	163	1483	811	1223
281	248	1168	82	293	1395	206	1181	117	35
479	906	1237	1016	457	523	903	905	1108	6
117	35	206	1181	992	398	1075	799	293	668
1331	567	816	893	680	269	945	851	1469	1543
1366	683	1440	1147	1334	985	579	1449	205	527
372	368	1406	907	525	891	1418	1144	182	277
945	186	1543	269	680	525	893	816	851	1331
885	1195	525	1045	1229	839	1086	966	186	1089
1275	1341	186	329	227	21	1048	1091	600	678
1500	874	1284	812	1430	470	527	1434	1273	171
1217	164	769	346	429	82	847	493	1277	715
1266	97	906	1162	1387	905	891	515	1016	1181
501	248	1341	992	117	600	1361	281	1091	35
1450	1212	368	1166	479	109	1237	646	779	6
523	163	13	1166	368	1341	281	479	811	1450
1168	352	194	392	533	132	1281	157	602	224
797	906	1016	457	1162	97	13	523	1387	905

Tabla A19. Resumen de la experimentación de EKCEAD ejecución final

712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229
1195	186	966	836	1089	712	669	1545	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	669	1545	1229	799
1089	1195	712	186	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	669	1545	1229	799
1089	1195	712	186	966	836	1545	669	1229	799
1195	186	966	836	1089	712	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	669	1545	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
1089	1195	712	186	966	836	1545	669	799	1229
1089	1195	712	186	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
1089	1195	712	186	966	836	1545	669	1229	799

712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	1545	669	1229	799
1089	1195	712	186	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229
1195	186	966	836	1089	712	669	1545	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	669	1545	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	1545	669	1229	799
1089	1195	712	186	966	836	1545	669	799	1229
1089	1195	712	186	966	836	1545	669	1229	799
1195	186	966	836	1089	712	669	1545	799	1229
1195	186	966	836	1089	712	1545	669	1229	799
1195	186	966	836	1089	712	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
1195	186	966	836	1089	712	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	1229	799
712	186	1195	1089	966	836	1545	669	799	1229