GENERACIÓN AUTOMÁTICA DE RESÚMENES DE MÚLTIPLES DOCUMENTOS CON UN ENFOQUE HIPERHEURÍSTICO BASADO EN ALGORITMOS MEMÉTICOS

DIANA PILAR ASTUDILLO MEDINA

Director: Dr. (c) MARTHA ELIANA MENDOZA BECERRA

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE SISTEMAS
GRUPO DE I+D EN TECNOLOGÍAS DE LA INFORMACIÓN
RECUPERACIÓN DE LA INFORMACIÓN
POPAYÁN, Abril 2013

Agradecimientos

Doy gracias a Dios por darme la fuerza, entendimiento y comprensión para enfrentar los momentos más difíciles.

A la Dr(c). Martha Eliana Mendoza Becerra y al Dr. (c). Carlos Alberto Cobos Lozada por su dedicación, tiempo, apoyo y enorme conocimiento para guiarme en este reto.

A mi familia, que me han acompañado en cada momento de mi vida, con su esfuerzo y aliento para que llegara a cumplir las metas propuestas.

A mis compañeros, amigos y educadores, quienes me acompañaron en este proceso, por su ánimo, colaboración y palabras de aliento.

Para finalizar, mis agradecimientos a la Universidad del Cauca institución que me forjó como persona, brindándome la oportunidad a través del programa de Ingeniería de Sistemas de realizar mi estudio de pregrado.

Tabla de Contenido

LISTA DE TABLAS

LISTA DE FIGURAS

Presentación

El crecimiento de información digital se presenta como una dificultad para su uso efectivo en la formación de conocimiento e investigación, ya que, es complicado leer toda la información que se necesita acerca de un tema específico. Por lo tanto, los usuarios gastan mucho tiempo y esfuerzo en examinar los documentos que requieren. Un área de investigación que intenta dar solución a este problema es la generación automática de resúmenes.

La generación automática de resúmenes de múltiples documentos puede ayudar a que se comprendan los principales temas de un documento sin tener que leer cada uno de los escritos originales. Desde hace muchos años, se han venido explorando diversos métodos para la creación automática de resúmenes para uno o más documentos y esto ha tomado gran importancia a medida que se aumenta la información disponible en la Internet.

Con el objetivo de explorar una nueva forma de generación automática de resúmenes de múltiples documentos, en el presente trabajo de grado se describe un nuevo algoritmo de generación automática de resúmenes de múltiples documentos obtenido con un enfoque Hiperheurístico basado en algoritmos meméticos.

En este documento se encuentran diferentes secciones que contienen la descripción de los conceptos teóricos y el procedimiento utilizado para el desarrollo del proyecto. A continuación se describe de manera general el contenido de esta monografía y su organización.

En el capítulo 1 se presenta la problemática que motivó el planteamiento de este proyecto, la justificación del desarrollo del mismo, los objetivos que se definieron y los principales resultados obtenidos.

El capítulo 2 se describe las bases teóricas que enmarcan el proyecto, teniendo en cuenta los conceptos básicos en el área de generación de resúmenes, investigaciones realizadas con respecto a los resúmenes para múltiples documentos, la forma como se evalúa el algoritmo planteado, la forma de representar los documentos para que puedan ser procesados por el algoritmo, descripción de la heurísticas de selección, cruce, reemplazo y búsqueda local, algoritmo memético.

En el capítulo 3 se presenta el entorno hiperheurístico propuesto, explicando las heurísticas utilizadas para la selección de alto nivel y bajo nivel utilizadas. Estas últimas incluyen esquemas de selección, cruce, reemplazo y búsqueda local; que permiten encontrar la mejor combinación del algoritmo memético para realizar la tarea de generación automática de resúmenes de múltiples documentos.

En el capítulo 4 se presenta el algoritmo memético para generación de resúmenes obtenido desde el enfoque Hiperheurístico, además el proceso de afinación realizado para la optimización de los parámetros del algoritmo memético.

En el capítulo 5 se presentan los resultados del enfoque hiperheurístico, la evaluación del algoritmo obtenido con este enfoque realizado a través de ROUGE y el análisis de los resultados obtenidos en el proceso de evaluación en comparación con los resultados de otros trabajos del estado del arte.

El capítulo 6 describe las conclusiones que se establecieron a partir de la experiencia adquirida en el desarrollo del proyecto y se proponen varias ideas de trabajo futuro para la continuidad del proyecto.

En el capítulo 7 se muestra la bibliografía y documentación empleada en la realización del proyecto.

Capítulo 1

1 INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

La información disponible en línea es un recurso esencial en muchas de las actividades de la investigación, en la actualidad, la información crece de forma exponencial y también la dificultad para lograr un efectivo aprovechamiento de ella. En consecuencia, los usuarios gastan mucho tiempo en examinar los documentos relacionados con su tema de interés. Para tratar de dar solución a este problema, desde hace muchos años se han venido realizando diversas investigaciones en el área de generación automática de resúmenes de múltiples documentos, que pretenden ofrecer al lector un resumen con los temas principales de los escritos, evitando de esta forma la lectura de cada uno de los documentos originales.

De esta forma, para la generación automática de resúmenes de un único o de múltiples documentos se han utilizado diversos enfoques. Entre los métodos para un documento se encuentran: los estadísticos [1], aprendizaje de máquina [2-5], conectividad de textos [6], grafos [7], reducción algebraica [8], métodos evolutivos entre ellos: los Algoritmos genéticos [9] (GA por sus siglas en inglés, Genetic Algorithm), Programación genética [10] (GP por sus siglas en inglés, Genetic Programing), Búsqueda armónica [11] (HS por sus siglas en inglés, Harmony Search) y Optimización de enjambres de partículas [12] (PSO por sus siglas en inglés, Particle Swarm Optimization). Así mismo, para múltiples documentos se tienen métodos basados en: conectividad de textos [13], grafos [14], reducción algebraica [15, 16], agrupamiento, modelos probabilísticos [17, 18] y evolutivos [19].

Con respecto a los modelos evolutivos, a pesar de que han obtenido buenos resultados en la generación de resúmenes de uno o múltiples documentos como ocurre con: un algoritmo genético que se compara con los mejores métodos de DUC[1] 2002 [9], programación genética comparada con un enfoque difuso y el modelo vectorial [10] y el algoritmo de optimización de enjambre de partículas (PSO) que utiliza el concepto de máxima cobertura y mínima redundancia de las oraciones de resumen [19] confrontado con los mejores métodos de DUC 2005 y 2007 tales como TranSumm, QEA, Content-term, entre otros. Aun así, falta explorar más en la generación automática de resúmenes para múltiples documentos con enfoques evolutivos que mejoren la calidad de éstos, obteniendo información más relevante de los escritos.

Por lo que se refiere al diseño de los modelos evolutivos, el proceso de seleccionar el esquema más apropiado de: selección, cruce y reemplazo, es muy complejo, ya que éste depende del problema en particular. El enfoque hiperheurístico es una alternativa para

[1]DUC (Document Understanding Conference), foro encargado de evaluar los sistemas de resumen.

realizar este proceso, en el cual, se usan heurísticas de bajo nivel para resolver el problema de generación automática de textos y heurísticas de alto nivel que permitan seleccionar y orientar el proceso de búsqueda en el espacio de las heurísticas de bajo nivel para el problema planteado, como se puede ver en [20], donde una heurística de alto nivel (selección por ruleta), en cada iteración selecciona heurísticas de bajo nivel más adecuadas para utilizarlas en la construcción de la solución al problema de reducción de los atributos. Por esto una hiperheurística puede ser vista como un método, que al darle una instancia particular y un número de heurísticas de bajo nivel, produce automáticamente una combinación adecuada de los componentes proporcionados para resolver eficazmente el problema dado [21].

Por otra parte, existe otro tipo de enfoque evolutivo llamado algoritmo memético (MA por sus siglas en inglés, Memetic Algorithm), que utiliza la optimización local, para refinar una solución y encontrar un óptimo en menos generaciones, además las soluciones generadas han sido de mayor calidad sobre varios dominios de aplicación que los tradicionales GAs [22].

Según lo anterior, en este proyecto de investigación basado en la eficiencia de las hiperheurísticas en otros tipos de aplicaciones, y que los algoritmos meméticos obtienen soluciones de mayor calidad con la optimización local, se propone un enfoque hiperheurístico para el problema específico de generación automática de resúmenes de múltiples documentos basado en algoritmos meméticos. Para esto, en el enfoque hiperheurístico se van a considerar heurísticas de bajo nivel que tengan en cuenta la selección de los esquemas de selección y cruce utilizados en los algoritmos genéticos y la búsqueda local usada en los algoritmos meméticos para mejorar la optimización de las soluciones candidatas, además con heurísticas de alto nivel que permitan seleccionar y orientar el proceso de búsqueda en el espacio de las heurísticas de bajo nivel.

El tipo de resumen que se va a generar en este proyecto tendrá las siguientes características de acuerdo a la taxonomía propuesta en [10]: extractivo, que selecciona las oraciones o frases del texto original; nivel de procesamiento superficial, que incluye características poco profundas (estadísticas y de similitud); genérico, porque va dirigido a una amplia comunidad de lectores; mono-lenguaje, solo se resumirán documentos en inglés; y múltiples documentos. Además la función objetivo que se utilizará estará basada en máxima cobertura y mínima redundancia, buscando que las oraciones del resumen sean relevantes al contenido de los documentos y se disminuya la redundancia de las oraciones [19].

Teniendo en cuenta lo anteriormente planteado, surge la siguiente pregunta de investigación ¿Es posible mejorar la calidad de los resúmenes generados automáticamente para múltiples documentos en el estado del arte (MCMR[2]), utilizando un enfoque hiperheurístico basado en algoritmos meméticos?

Por lo tanto, en el presente trabajo de investigación se propone un algoritmo de generación automática de resúmenes extractivos de múltiples documentos con un

[2] (MCMR por sus siglas en inglés Maximum Coverage and Minimum Redundant), Máxima Cobertura y Mínima Redundancia.

enfoque hiperheurístico basado en algoritmos meméticos, que permita seleccionar las mejores heurísticas de bajo nivel de los algoritmos genéticos; y búsqueda local.

1.2 JUSTIFICACIÓN

La generación automática de resúmenes se comienza a investigar a inicios de la década de los 50s, sin embargo desde los 90s ha tomado mayor importancia debido al aumento considerable de publicaciones científicas y no científicas. Esta enorme cantidad de documentos existente y almacenada en diferentes formatos ha dado origen a la creación de nuevos métodos dentro de la generación automática de resúmenes, cuyo enfoque es reducir el tiempo y esfuerzo que el usuario debe invertir en leer un documento de su interés.

Debido a los retos que presenta la tarea de generación automática de resúmenes, conseguir resúmenes igual que los haría un humano es aún un tema de investigación, debido a que no se ha logrado resultados óptimos en la generación de resúmenes de múltiples documentos. Por lo tanto, es necesario seguir investigando en técnicas para identificar la información más relevante de los documentos y presentarla a manera de resumen.

El aporte principal de este proyecto es proponer un algoritmo memético para generación automática de resúmenes de múltiples documentos obtenido desde un enfoque Hiperheurístico; para su planteamiento fue necesario tener en cuenta varios aspectos importantes como son: investigación de las hiperheurísticas, aplicación de diferentes configuraciones en la etapa de ejecución de la hiperheurística, adaptación de los esquemas de selección de alto nivel para seleccionar los esquemas de bajo nivel, lo mismo que adaptación de los esquemas de bajo nivel para el problema de generación automática de resúmenes de múltiples documentos y el afinamiento del algoritmo memético obtenido con el enfoque hiperheurístico para un mejor rendimiento del mismo.

Las herramientas tecnológicas para desarrollar el proyecto[3] fueron seleccionadas de acuerdo a la experiencia en su manejo por el Grupo de Tecnologías de la Información (GTI) durante los últimos años, sobre todo en proyectos de trabajo de grado exitosos relacionados con Recuperación de la Información. Además dada la disponibilidad del software, documentación y materiales de aprendizaje que se tiene en la Universidad del Cauca.

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

Proponer un algoritmo de generación automática de resúmenes extractivos[4] de múltiples documentos con un enfoque hiperheurístico basado en algoritmos meméticos.

[3] Herramientas tecnológicas: Microsoft Visual Studio 2010, Microsoft Project 2010, Microsoft Office 2010.

[4] Extractivos consiste en la selección de oraciones del texto principal.

1.3.2 OBJETIVOS ESPECÍFICOS

- Definir un entorno hiperheurístico[5] para generación automática de resúmenes extractivos de múltiples documentos basado en algoritmos meméticos, que utilice una función objetivo basada en máxima cobertura y mínima redundancia; además como heurísticas de alto nivel ruleta y torneo probabilístico, y como heurísticas de bajo nivel: esquemas de selección y cruce[6] de individuos, los algoritmos de búsqueda local iterada y de vecindad, y los esquemas de reemplazo del peor individuo y competencia restringida.

- Implementar un prototipo basado en el entorno hiperheurístico definido, que permita determinar el algoritmo memético que obtenga los mejores resultados para el problema de generación automática de resúmenes extractivos de múltiples documentos.

- Evaluar el algoritmo memético obtenido y compararlo con otros algoritmos evolutivos de generación automática de resúmenes extractivos de múltiples documentos del estado del arte, utilizando medidas de ROUGE y conjuntos de datos de DUC.

1.4 RESULTADOS OBTENIDOS

- Código fuente del entorno hiperheurístico para la selección de la mejor combinación de los esquemas de bajo nivel y código fuente del algoritmo memético obtenido y evaluación de la calidad del mismo.

- Artículo: Algoritmo memético para generación automática de resúmenes de múltiples documentos obtenido desde un enfoque Hiperheurístico.

- Monografía del trabajo de grado. Corresponde al presente documento, donde se describe lo que dio origen a la realización del proyecto, las bases teóricas que lo enmarcan, la definición y características del entorno hiperheurístico propuesto, la configuración del algoritmo memético propuesto, los resultados obtenidos, los aportes más sobresalientes, las conclusiones y las recomendaciones para el desarrollo de futuras investigaciones en el área.

[5] Un entorno hiperheurístico permite la selección de heurísticas de bajo nivel por medio de heurísticas de alto nivel y aplicación de las heurísticas seleccionadas en el problema específico.

[6] Por restricciones del alcance del proyecto, se tomarán máximo dos esquemas de selección y dos de cruce, de los esquemas más ampliamente usados en los algoritmos genéticos y meméticos.

Capítulo 2

2 CONTEXTO TEÓRICO

2.1 GENERACIÓN AUTOMÁTICA DE RESÚMENES

2.1.1 El resumen

Según Spärck Jones [23], un resumen es “una transformación reductiva del texto fuente al resumen a través de la condensación mediante la selección y/o generalización de lo que es importante en el texto original”. Por lo tanto, se puede decir que un resumen es una breve representación de uno o varios documentos que contiene la información importante de los documentos originales; y la generación automática de resúmenes consiste en la técnica por la cual un computador crea un resumen automáticamente.

2.1.2 Taxonomía de los resúmenes

Se pueden considerar diferentes taxonomías de resúmenes [24], entre las más citadas se encuentran:

- De acuerdo a la forma.
  a. Extracción, selecciona las oraciones o frases del texto original.
  b. Abstracción, contiene oraciones que no se encuentran en el texto original, por lo que es un tarea de investigación.

- Nivel de procesamiento.
  a. Nivel superficial, representan al documento con características poco profundas como: términos específicos de una consulta de usuario, términos estadísticamente relevantes, frases clave, entre otros.
  b. Nivel Profundo, requieren de técnicas como el procesamiento de lenguaje natural para su construcción.

- Propósito del resumen.
  a. Indicativos, dan información abreviada sobre los temas principales de un documento, su objetivo es ayudar a los usuarios a leerlo o no.
  b. Informativos, cubren toda la información importante del texto original, actuando como sustituto para que el lector no tenga necesidad de leer todo el documento.
  c. Críticos, evalúan el tema o contenido del texto de entrada, expresando el punto de vista de la persona que realiza el resumen.

- Audiencia a la que va dirigido el resumen.
  a. Genéricos, no dependen de la audiencia a la que va dirigido.
  b. Basados en consultas, los resultados se basan en una pregunta que ha realizado un usuario.
  c. Enfocados en el usuario o en temas, se adaptan a los intereses de los usuarios o solo se enfocan en un tema particular.
- Cantidad de documentos que se procesan.

a. Un documento.
b. Múltiples documentos.

- El lenguaje del documento.
  a. Monolenguaje, procesan un texto escrito en un solo idioma.
  b. Multilenguaje, si el texto original está escrito en diferentes idiomas.

- Tipo de documento.
  a. Artículos científicos.
  b. Reportes.
  c. Noticias, entre otros.

2.1.3 Algoritmos para la generación automática de resúmenes extractivos para múltiples documentos

2.1.3.1 Métodos basados en conectividad de textos

Los métodos basados en conectividad de textos exploran las propiedades cohesivas para comprender las relaciones entre las expresiones del texto, por ejemplo, las expresiones anafóricas que se refieren a partes mencionadas anteriormente del texto deben conocer sus antecedentes con el fin de ser comprendidos, si una frase se extrae sin el contexto anterior, el resumen puede ser difícil de entender. Uno de los métodos que explora éstas propiedades es el modelo basado en cadenas léxicas [6]. Este modelo consiste en realizar un procesamiento profundo del texto, utiliza la herramienta WordNet para determinar los significados de las palabras y las relaciones de ellas. Para el tratamiento de múltiples documentos Chen et al [13], proponen un método de generación de resúmenes de múltiples textos basado en cadenas léxicas, el algoritmo construye las cadenas léxicas para cada documento por medio de una base de datos de conocimiento, identifica las cadenas más fuertes, las oraciones significativas son extraídas de cada documento, se mezclan, se ordenan y para el resumen se seleccionan las cadenas con mayor puntuación y con la menor similitud hacia las oraciones ya seleccionadas. Una de las limitaciones del método propuesto es que no hay ningún control sobre la longitud del resumen.

Mieskes et al [25], proponen un método en que combinan el filtrado de documentos, ranking de oraciones utilizando cadenas léxicas y algoritmos de grafos que coincidan con el tema, además de varias capas de anotación con la herramienta de anotación MMAX2, para la generación de resúmenes de múltiples documentos. Para el manejo de redundancia de oraciones, se toma la oración con puntaje más alto de similitud, luego la siguiente, hasta que se alcance un límite de 250 palabras. Esta técnica de resumen de extracción se basa en varias etapas de filtrado, tanto en el documento como en las oraciones. El primer filtro es el nivel del documento, del conjunto de 30 documentos por tema, seleccionan un número pequeño en función de su coherencia con el tema correspondiente; la reducción de documentos permite realizar un análisis semántico más profundo. El siguiente filtro es el nivel de la oración, las oraciones se obtienen basadas en la información de las cadenas léxicas arraigados en los sustantivos y los verbos de este tema. Este paso de filtrado en el documento reduce el número de oraciones redundantes con respecto al tema.

2.1.3.2 Métodos basados grafos

Estos enfoques básicamente representan los documentos en forma de grafo para capturar los conceptos centrales [26]. Estos enfoques representan el texto como una red compleja en la que los nodos representan cada una de las unidades textuales en las que se divide el texto y las aristas representan algún tipo de relación entre estas unidades. La idea en este tipo de enfoques es la emergencia en la red de grupos de unidades que guardan estrecha relación entre sí y que determinan la información relevante del documento.

Mihalcea [14], propone un método que utiliza los mismos principios del ordenamiento basado en grafos que fueron aplicados en generación automática de resúmenes de un solo documento [7]. Primero se realiza el resumen de cada documento y luego se resumen los resúmenes de los documentos individuales utilizando el mismo método de grafos. Con respecto a la alta similitud entre las oraciones, manejan un umbral máximo de similitud de oraciones.

Gunes and Dragomir [27], proponen un método estocástico basado en grafos para calcular la importancia relativa de las unidades de texto en el procesamiento del lenguaje natural. Aplica el análisis de grafos y tiene en cuenta la influencia de otras frases, lo que permite un mejor panorama de las relaciones entre las frases. Primero construye un grafo de todas las frases candidatas, donde los nodos son las oraciones y las aristas son los valores de similitud del coseno. Dos oraciones candidatas están conectadas con una arista si la similitud entre ellos está por encima de un umbral. El sistema busca las frases más centrales del grafo mediante la realización de un recorrido aleatorio.

Los métodos basados en grafos tienen la ventaja de ser altamente independientes del lenguaje y cuando se pretende obtener una mayor precisión no resulta tan compleja la adaptación a un idioma en particular, sin embargo el método no tiene en cuenta las relaciones semánticas entre las palabras por ser independiente del lenguaje a menos que se lleve a un idioma o a un tema específico. Además, cuando la dimensionalidad es alta (muchas frases o términos en el documento) el costo en términos de ejecución se incrementa considerablemente.

2.1.3.3 Métodos basados en reducción Algebraica

Varios métodos se han propuesto con esta técnica. El más utilizado es el Análisis Semántico Latente (LSA), es un método para la extracción y representación del significado contextual de las palabras mediante cálculos estadísticos aplicados a un amplio corpus de texto. Además permite la creación de vectores multidimensionales para el estudio de las relaciones existentes entre palabras y párrafos.

Hachey et al [15], proponen un enfoque de generación automática de resúmenes de múltiples documentos orientado a consulta basado en análisis semántico latente (LSA por sus siglas en inglés, Latent Semantic Analysis) y en relevancia máxima marginal (MMR por sus siglas en inglés Maximal Marginal Relevance). Se aplica LSA y se tienen en cuenta las siguientes restricciones: Tiempo (preservar el orden temporal de los eventos), Secuencia (preservar el orden original de las oraciones), Grupos (grupos con oraciones similares), Contexto (recrear el contexto precedente original). El algoritmo es determinístico y optimiza localmente las oraciones extraídas, determina la oración con mayor puntaje, y la mueve al final del resumen. Repite el proceso hasta que todas las oraciones extraídas sean insertadas.

Steinberger y Křišťan [16], proponen una extensión de LSA, primero se crea una matriz de términos por oración que incluye todas las oraciones del conjunto de documentos, el puntaje se calcula de la misma forma que se hace para un único documento, y se seleccionan las oraciones con mayor puntuación para el resumen. Para evitar la redundancia, antes de incluir una oración en el resumen, se revisa si ya existe una oración similar, que debe estar cerca a la consulta del usuario. Este método favorece las oraciones largas, porque estas probablemente contendrán más términos importantes que una corta, por esto, se divide la puntuación de la oración por el número de términos *lk*, donde *lk* es el coeficiente de longitud.

2.1.3.4 Métodos basados en agrupamiento y modelos probabilísticos

Radev et. al [28], proponen un generador de resúmenes llamado MEAD , que usa los centroides de grupo producidos por un sistema de detección y seguimiento de tópicos antes de generar el resumen, identifican los artículos sobre un evento, este proceso es llamado Detección y seguimiento de tópico (TDT). MEAD utiliza MMR para eliminar la redundancia en el resumen y recibe como entrada *n* oraciones de un grupo de documentos y la tasa de compresión *r*, y genera como salida *n*r* oraciones del grupo con los puntajes más altos. El puntaje de cada oración tiene en cuenta características como: valor del centroide, valor posicional y solapamiento con la primera oración. A este valor se le resta una penalidad por redundancia. Cada documento de cada grupo se califica y se ordena la oración de cada documento de acuerdo al puntaje.

Hennig [18], propone un método de generación de resúmenes orientado a consulta basado en análisis semántico latente probabilístico (PLSA por sus siglas en inglés, Probabilistic Latent Semantic Analysis), el cual permite representar las oraciones y las consultas como distribuciones de probabilidad sobre tópicos latentes. PLSA permite modelar los documentos como una mezcla de tópicos latentes. El resumen se produce en tres pasos: 1) Crean la matriz de términos por oración y se entrena el modelo PLSA sobre esta matriz; 2) Calculan las diferentes características a nivel de oración basado en la similitud de las distribuciones de las oraciones y de la consulta sobre los tópicos latentes; 3) Calculan el puntaje de la oración como la combinación lineal de los puntajes de las características y se ordenan las oraciones de acuerdo al puntaje, luego se seleccionan las oraciones con el mejor puntaje; estos puntajes son actualizados por medio de una penalidad, de acuerdo a la similitud de cada oración con el resumen actual.

2.1.3.5 Métodos basados en modelos evolutivos

Los algoritmos evolutivos [29] surgieron como simulación de procesos de evolución natural, tuvieron su origen en el año 1960 y fueron introducidos por John Holland quien incorporó métodos de selección natural y supervivencia a la resolución de problemas de inteligencia artificial . Un algoritmo evolutivo consta de una función objetivo que permite evaluar las soluciones candidatas dando mecanismos de selección que permitan crear nuevas soluciones al problema que se desea resolver.

Con respecto a la generación de resúmenes de múltiples documentos basada en algoritmos evolutivos se puede encontrar los siguientes trabajos.

Rasim et al [19], proponen un modelo de generación automática de textos sin supervisión con máxima cobertura y mínima redundancia (MCMR), para generar un resumen mediante la extracción de frases más destacadas de los documentos dados. Este modelo

intenta optimizar tres propiedades: 1) relevancia: el resumen debe contener oraciones relevantes para el usuario, 2) redundancia: los resúmenes no deben contener varias unidades de texto que comuniquen la misma información y 3) longitud: el resumen es limitado. Para la optimización del problema se utilizan los algoritmos de: ramificación y poda, y optimización por enjambre de partículas binarias. Los resultados experimentales sobre el conjunto de datos DUC2005 y DUC2007 mostraron que esta propuesta supera a los sistemas de referencia.

Bossard et al [30], proponen un modelo que realiza una combinación de un sistema de generación de resúmenes de múltiples documentos con un algoritmo genético en donde hacen agrupamiento y el método de selección se basa en la centralidad local para extraer una oración por grupo, adicionalmente hacen uso de un algoritmo genético para realizar la optimización de los catorce parámetros utilizados para la creación del resumen.

2.1.4 Evaluación de la Calidad de los Resúmenes

En los primeros métodos la evaluación de los resúmenes generados de manera automática se realizó de una forma manual [31], tarea que fue difícil debido a que no había un resumen ideal para realizar la comparación. Por lo tanto, requirió del juicio de una persona experta capaz de detectar la coherencia, consistencia gramatical, legibilidad y contenido del documento. Esto conllevo a que se presentara mucha subjetividad dependiendo de la opinión de los jueces, y a necesitar de un esfuerzo humano considerable.

2.1.4.1 Tipos de Evaluación

Los métodos básicos para evaluar la calidad de los resúmenes son: la evaluación intrínseca que no tiene en cuenta a la audiencia a la que va dirigida; y extrínseca que tiene en cuenta a la audiencia a la que va dirigida [32].

2.1.4.1.1 Evaluación Intrínseca

La evaluación intrínseca mide la calidad del resumen sin tener en cuenta a la audiencia a la que este va dirigido, da mayor peso a aspectos como la coherencia o lo informativo del resumen generado. Esta evaluación se realiza comparando los resúmenes ideales, uno por cada documento o por cada conjunto de documentos en el caso de resumes de múltiples documentos. Este tipo de evaluación se puede hacer por medio de medidas estándar de Precisión y Recuerdo [33], y medidas de ROUGE [34]. Dado que no existe un resumen “perfecto", algunas investigaciones para la evaluación del resumen generado utilizan más de un resumen realizado por humanos para cada documento de prueba o por cada conjunto de documentos de prueba, y promedian el puntaje obtenido por el sistema a través del conjunto de resúmenes ideales.

2.1.4.1.2 Evaluación Extrínseca

La evaluación extrínseca de un resumen tiene su enfoque en el usuario al que va dirigido el resumen, teniendo más en cuenta la utilidad que este puede tener sobre ese usuario que su calidad como resumen. Por lo tanto, mide la eficiencia y aceptación de los resúmenes generados en alguna tarea, por ejemplo la evaluación de relevancia o comprensión de la lectura [33]. Otras tareas que se pueden medir son la recopilación de la información en una colección de documentos de gran tamaño, el esfuerzo y tiempo requerido para enviar a editar el resumen generado a máquina para algún propósito

específico. Éste tipo de evaluación requiere del esfuerzo de muchas personas y más aún cuando se trata de resumir muchos documentos extensos. Se han propuesto diversos escenarios para la evaluación extrínseca como el Juego de Shannon, el Juego de la Pregunta, el juego de la Categorización o Clasificación y Asociación de claves [33].

2.1.4.2 Evaluación con Rouge

Para la evaluación se hizo uso de la herramienta ROUGE 1.5.5 [35] (Recall-Oriented Understudy for Gisting Evaluation), que contemplan medidas creadas especialmente para evaluar la calidad de resúmenes de texto generados automáticamente; estas medidas se empezaron a utilizar en DUC, el cual es un foro que se viene realizando desde el año 2000, encargado de evaluar los sistemas generadores de resúmenes, que busca lograr que se defina un estándar en el modo cómo se realiza la evaluación de estos sistemas; los integrantes de este foro pueden entrenar y evaluar sus sistemas gracias a la gran cantidad de conjuntos de documentos que aquí se encuentran.

Esta herramienta permite calcular diversas medidas, principalmente ROUGE-N, ROUGE-L, ROUGE-W, ROUGE-S y ROUGE-SU. La diferencia entre ellas está en la cantidad o en la forma como se tomen los n-gramas.

- ROUGE-N: Mide la superposición de n-gramas de palabras entre el resumen del sistema y los resúmenes de referencia. Se calcula como se muestra en la Ecuación 1:

$$\text{ROUGE} - \text{N} = \frac{\sum_{S \in \{ResumenesModelo\}} \sum_{grama_n \in S} Conteo_{n-gramas\ que\ coinciden}}{\sum_{S \in \{ResumenesModelo\}} \sum_{grama_n \in S} Conteo_{n-gramas}} \quad \text{Ecuación 1}$$

  Donde *n* es la longitud del n-grama y $Conteo_{n-gramas\ que\ coinciden}$ es el número máximo de n-gramas co-ocurrentes en el resumen candidato y el conjunto de resúmenes modelo. El denominador de la ecuación es el número total de n-gramas ocurrentes en el resumen modelo.

- ROUGE-L: se basa en obtener la subsecuencia común más larga (LCS por sus siglas en inglés, Longest Common Subsequence) entre dos textos.

  Tomando una oración del resumen como una secuencia de palabras, se propone LCS para estimar la similitud entre dos resúmenes, *X* de longitud *m* y *Y* de longitud *n* asumiendo *X* como la oración del resumen modelo y *Y* como la oración candidata del resumen. El recuerdo se calcula como se muestra en la Ecuación 2:

$$R_{lcs} = \frac{\sum LCS\ (X, Y)}{m} \quad \text{Ecuación 2}$$

  Donde *m* es la longitud del resumen modelo y LCS es la longitud de la sub-secuencia común más larga de *X* y *Y*.

- ROUGE-W: es similar a ROUGE-L con la diferencia de que utiliza una modificación de la LCS básica. Dicha modificación consiste en memorizar los tamaños de los emparejamientos consecutivos y quedarse con el mayor.

- ROUGE-S (Skip-Bigram Co-Occurrence Statistics): Es cualquier par de palabras en el orden de la oración, permitiendo espacios arbitrarios. *Skip-Bigram Co-Occurrence Statistics* mide la superposición de los Skip-Bigrams entre el resumen candidato y el resumen modelo.

  Dados dos resúmenes X de longitud *m* y Y de longitud n, asumiendo que X es el resumen modelo y Y el resumen candidato se puede medir el Recuerdo mediante ROUGE-S como se muestra en la:Ecuación 3.

  $$R_{skip2} = \frac{SKIP2(X,Y)}{C(m,2)}$$ Ecuación 3

  Donde SKIP2(X,Y) es el número de Skip-Bigram que coinciden entre X y Y y C(m,2) es la cantidad de Skip-Bigram obtenidos de X.

  Por ejemplo, dados:

  X: police killed the gunman
  Y: police kill the gunman

  Se pueden obtener los siguientes Skip-Bigram:

  Skip-Bigram(X) = {police killed, police the, police gunman, killed the, killed gunman, the gunman}
  Skip-Bigram(Y) = {police kill, police the, police gunman, kill the, kill gunman, the gunman}

  Los Bigramas en común entre X y Y son: {police the, police gunman, the gunman}, entonces el valor de R_skip2=3/6=0,5

  El resultado obtenido con R_skip2 es más intuitivo que ROUGE-L, la ventaja es que no requiere de coincidencias consecutivas aunque aún se conserva cierta sensibilidad al orden de las palabras. R_skip2 cuenta todas las concordancias de pares de palabras en orden mientras que ROUGE-L solo cuenta la sub-secuencia común más larga.

- ROUGE-SU (Extensión de ROUGE-S): Un problema potencial de ROUGE-S es que no da ningún crédito a las oraciones candidatas si la oración no tiene ninguna pareja de palabras co-ocurrentes entre el resumen modelo y el resumen candidato. Por ejemplo dado el resumen modelo X = {police killed the gunman} y el resumen candidato Y = {gunman the killed police} en ROUGE-S tendrían una similitud de cero, por lo que podría ser conveniente calcular la co-ocurrencia de palabras no a través de bigramas sino de unigramas.

2.2 REPRESENTACIÓN DE LOS DOCUMENTOS

2.2.1 Modelo vectorial

Consiste en representar cada unidad de texto (oraciones[7]) como un vector de términos ponderados. El modelo vectorial está basado en que cada oración de la colección está representada por un vector n-dimensional (n es la cardinalidad del conjunto de términos de indexación elegido para toda la colección de oraciones), en el que cada componente representa el peso del término asociado a esa dimensión. Este peso representa un estimado estadístico de la utilidad del término como descriptor del documento, es decir, de la utilidad de distinguir esa oración del resto de unidades de la colección. Un término recibe un peso de 0 en las oraciones en las cuales éste no ocurre. Normalmente los términos muy comunes y los poco frecuentes son eliminados.

Dado un documento D formado por N unidades textuales, entonces una oración es representada según el modelo vectorial, como un vector $\overrightarrow{s_j} = (w_{1j}, w_{2j}, \ldots, w_{nj})$ donde n es el número total de términos de D y w_{nj} representa la importancia de término dentro de la oración.

2.2.2 Esquemas de Pesado de Términos

Existen diferentes maneras de asignar el peso a los términos [36], entre ellas está la técnica de pesado *booleano* donde los pesos $w_i \in \{0,1\}$ indican la presencia o ausencia del término t_i en el documento, otra es la técnica de *frecuencia de los términos* [37] que indica el número de veces que el término t_k aparece en el documento, denotado $TF_{t,d}$.

La técnica usada en este trabajo (ponderación por frecuencia relativa *TF-ISF)*, esta es una mezcla de *TF* con la *frecuencia inversa de la oración (ISF[8])* que tiene que ver con la poca frecuencia de un término en la colección de oraciones. El cálculo de los pesos se hace de la siguiente manera.

$$W_{t,s} = TF_{t,s} \times ISF_t \qquad \text{Ecuación 4}$$

Donde:

$$TF_{t,s} = \frac{Freq_{t,s}}{Max\ Freq_s} \qquad \text{Ecuación 5}$$

$$ISF_t = \log\left(\frac{N}{n_t}\right) \qquad \text{Ecuación 6}$$

Donde $TF_{t,s}$ es la frecuencia del término *t* en la oración *s,* ISF_t es la frecuencia invertida del término *t*, $Freq_{t,s}$ es la frecuencia del término *t* en la oración *s*, $Max\ Freq_s$ es la

[7] Oración: En este caso es una oración simple o compuesta de varias oraciones

[8] ISF:Inverse Sentence Frequency"

máxima frecuencia de términos la oración *s*, *N* es la cantidad de frases en la colección y n_t es la cantidad de veces que aparece el término *t* en todas las oraciones.

2.2.3 Procesamiento de Múltiples Documentos

La representación de múltiples documentos se hace basado en el modelo de espacio vectorial y de acuerdo a [38], donde $D = \{d_1, d_2, \dots, d_n\}$ es el conjunto de documentos y *n* es el número total de documentos; por facilidad se representa la colección de documentos como un conjunto de todas las oraciones de los documentos en la colección, es decir $D = \{s_1, s_2, \dots, s_m\}$ donde *m* es el número total de oraciones de la colección de documentos y *s* es la colección de términos $s_{i,} = \{t_{1,i}, t_{2,i}, \dots, t_{k,i}\}$. El objetivo es obtener un subconjunto de D con las oraciones que satisfagan dos factores para la generación de un buen resumen. Los factores serán definidos más adelante. Cada oración s_i es representada como un vector con los pesos de los términos $\overrightarrow{s_i} = \{w_{i,1}, w_{i,2}, \dots, w_{i,m}\}$. Donde *m* es el número de términos en la colección de documentos, $w_{i,k}$ es el peso del término t_k en la oración s_i. El componente w_{ik} se define usando la matriz de términos por documento *tf-isf,* como se explica en la sección 2.2.2.

2.2.4 Medida de similitud

Una forma de determinar qué tanto se parecen los documentos u oraciones es estableciendo una relación entre ellos a través de la comparación de los términos que los componen. Para realizar la comparación se puede hacer uso de distintas medidas de similitud, las cuales juegan un rol importante en el área de generación de resúmenes. Se debe tener en cuenta que para aplicar alguna medida de similitud es necesario definir una manera de representación de los documentos donde esta pueda ser aplicada. Para el caso de este proyecto, el modelo de representación vectorial. A continuación presento la medida utilizada en este problema.

2.2.4.1 Medida de cosenos

Esta medida es ampliamente usada en generación de resúmenes debido a su sensibilidad, a la importancia relativa de cada palabra, y ha sido usada en [18, 39-42], entre otros. La idea básica es medir el ángulo entre el vector de $\overrightarrow{s_i}$ de $\overrightarrow{s_j}$, para hacerlo calculamos:

$$SC(s_i, s_j) = \frac{\sum_{k=1}^{t} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{t} w_{jk}^2 \ \sum_{k=1}^{t} w_{jk}^2}} \qquad \text{Ecuación 7}$$

Donde *k* va de 1 al número total de términos del vocabulario *t,* W_{ik} indica el peso del término *k* en la oración s_i y W_{jk} el peso del término *k* en la oración s_j. Este peso fue obtenido con la Ecuación 4.

La medida de cosenos es una función trigronométrica que mide el coeficiente de similitud entre dos conjuntos o dos oraciones, representados en un espacio vectorial. Mide el angulo *(0º<beta<90º)*, que indica que tan cercano esta el uno del otro, en terminos de la dimensionalidad [43]. Entre mas pequeño sea el angulo mayor será la similitud, al aplicar la formula, el valor estará entre 0 y 1 siendo 1 el mayor grado de similitud entre dos oraciones.

2.3 HIPERHEURÍSTICAS

2.3.1 Definición

Es importante primero definir que es una heurística, es un método basado en la experiencia que se puede utilizar para obtener soluciones óptimas a problemas complejos, como por ejemplo, el vendedor viajero, planeación de tareas [44]. Las hiperheurísticas son un método de búsqueda que está motivada por el objetivo de automatizar el proceso de selección o la combinación más simple de heurísticas para resolver problemas difíciles de búsqueda computacional [45]. Una breve definición de los métodos hiperheurísticos es “heurísticas para elegir heurísticas”. La hiperheurística aplica la heurística correcta durante el proceso de resolución de problemas de acuerdo con el estado actual de la solución [46].

2.3.2 Esquema General

Por lo que se refiere al esquema de una hiperheurística, en la Figura 1 se da a conocer la esquematización de ésta, de acuerdo a la propuesta realizada en [47]. Donde en cada etapa de la búsqueda de las heurísticas de bajo nivel a utilizar en el dominio del problema, utiliza la información de los resultados anteriores de las heurísticas de bajo nivel para seleccionar las nuevas. La selección se hace a menudo utilizando una función de selección y este proceso continúa hasta que una condición de parada esté satisfecha, la mejor solución se determina basándose en el valor de la función de coste. Cabe destacar que en el controlador de la hiperheurística se maneja la función de selección, y en las medidas de desempeño se emplean los resultados de la función objetivo para conocer qué tan buena es la combinación de heurísticas de bajo nivel.

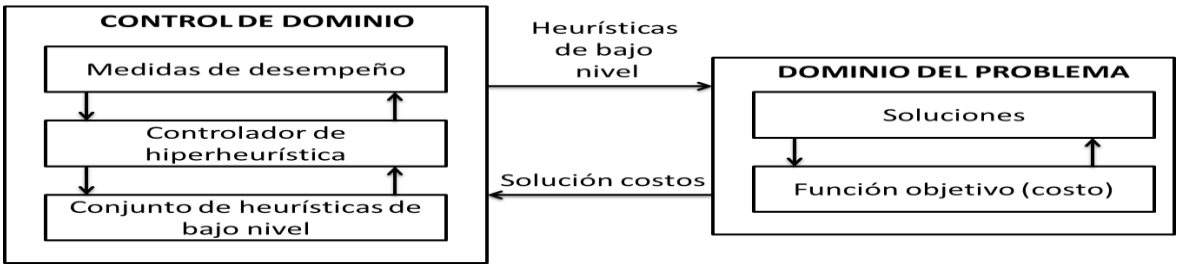

Figura 1. Esquema de una hiperheurística

2.3.3 Tipos de Hiperheurísticas

Existen diversos mecanismos de selección para las hiperheurísticas los cuales se pueden clasificar en:

2.3.3.1 Hiperheurísticas con aprendizaje o sin aprendizaje

En este grupo se incluyen las hiperheurísticas que emplean diversas técnicas de aprendizaje de los resultados que han aportado cada heurística de bajo nivel a lo largo del proceso de búsqueda. Una hiperheurística con aprendizaje selecciona una heurística de bajo nivel en cada iteración basada en la eficacia acumulada de cada heurística desde el inicio de la ejecución hasta el estado actual. Y las hiperheurísticas sin aprendizaje seleccionan las heurísticas de acuerdo a una secuencia predeterminada [48].

2.3.3.2 Hiperheurísticas constructivas o con búsqueda local

Las hiperheurísticas constructivas construyen una solución incremental de forma adaptativa seleccionando la heurística; las hiperheurísticas con búsqueda local empiezan a partir de una solución inicial y seleccionan iterativamente a partir de un conjunto de estructuras vecinas.

2.3.3.3 Hiperheurísticas basadas en selección aleatoria

Estas hiperheurísticas basadas en selección aleatoria han sido el enfoque de selección más antiguo, porque es el mecanismo más simple y fácil de aplicar. En este mecanismo de selección se elige al azar una heurística de bajo nivel y siempre se aplica, así no produzca alguna mejora a la solución actual del problema [49].

La desventaja de un enfoque aleatorio es que la calidad de la solución obtenida depende de la posibilidad de seleccionar una buena secuencia de heurísticas de bajo nivel. Con el fin de no quedar atrapados en regiones pobres del espacio de búsqueda se han realizado modificaciones, consisten en afectar la regla de no aceptar siempre todos los movimientos malos. Cowling et al. en [50, 51], compara diferentes versiones del mecanismo de selección aleatorio aplicado al problema de programación de reuniones. Los mecanismos son: aleatorio simple, escoge aleatoriamente la heurística a aplicar en cada iteración; aleatorio descendente, selecciona una heurística aleatoriamente y le aplica repetidamente conforme el resultado se va mejorando y se detiene hasta que ningún otro movimiento produzca mejora; permutación aleatoria, crea una permutación inicial de las heurísticas y en cada iteración aplica la heurística por cada una de las posiciones de la permutación; permutación aleatoria descendente, es similar al anterior sólo que la heurística se aplica hasta que el resultado de aplicar una cierta heurística deje de mejorar.

2.3.3.4 Hiperheurística codiciosa

Las hiperheurísticas codiciosas seleccionan y aplican en cada iteración la heurística de bajo nivel que produce la mayor mejora en el valor de la función objetivo, o aquella que produzca el menor deterioro si no se mejora la solución previa. Hay dos estrategias codiciosas: una acepta sólo mejoras de la heurística de bajo nivel mientras que la segunda acepta tanto movimientos que mejoran como aquellos que no mejoren a la solución actual. La segunda estrategia es mejor porque impide a la hiperheurística detenerse en un óptimo local.

La desventaja del mecanismo codicioso es su limitada capacidad para explorar con eficacia el espacio de búsqueda, dejando regiones con soluciones potencialmente fuertes no visitadas [52].

2.3.3.5 Hiperheurísticas basadas en metaheurísticas

Una metaheurística es un algoritmo de búsqueda para dar una solución a problemas complejos, que se ha aplicado de forma exitosa en problemas difíciles de optimización del mundo real. Varios enfoques metaheurísticos y sus híbridos han sido probados como mecanismos de selección de las hiperheurísticas en los últimos años y se les suele llamar hiperheurísticas basadas en metaheurísticas [53].

Las hiperheurísticas basadas en algoritmos genéticos han servido como punto de partida para crear nuevas hiperheurísticas donde su mecanismo de selección esta inspirados en otras metaheurísticas.

2.4 HEURÍSTICAS DE SELECCIÓN, CRUCE, REEMPLAZO Y BÚSQUEDA LOCAL

2.4.1 Selección

Los algoritmos de selección son los encargados de escoger que individuos van a reproducirse y cuáles no. A continuación se describen los métodos más conocidos [54] [55] [56].

2.4.1.1 Selección por ruleta

Propuesto por DeJong, es posiblemente el método más utilizado desde los orígenes de los Algoritmos Genéticos, y el más simple de selección. A cada uno de los individuos de la población se le asigna un valor entre cero y uno, de tal forma que la suma de todos los porcentajes sea la unidad. Generalmente la población está ordenada con base en la función objetivo por lo que los valores más altos se encuentran al inicio. Para seleccionar un individuo basta con generar un número aleatorio del intervalo [0..1], recorrer los individuos de la población y acumular sus porcentajes hasta que la suma exceda el número aleatorio, y luego devolver el individuo situado en esa posición de la población [57].

2.4.1.2 Selección por torneo

La selección se realiza con base en comparaciones directas entre individuos, hay dos versiones de selección por torneo: determinística y probabilística [58].

2.4.1.2.1 Selección por torneo determinístico

Consiste en seleccionar al azar *p* individuos de la población, de esos *p* individuos aquel que tenga mayor función objetivo gana el torneo y se convierte en uno de los padres, este proceso se repite para seleccionar el segundo de los padres que se incluirá.

2.4.1.2.2 Selección por torneo probabilístico

Se diferencia de la anterior selección en el paso de seleccionar el ganador del torneo. No siempre escoge el mejor, genera un número aleatorio *r* entre cero y uno; si el número generado es mayor que un parámetro fijado (para todo el proceso evolutivo) se escoge el más alto en caso contrario el menos apto.

2.4.1.3 Selección uniforme

Consiste en seleccionar de forma aleatoria un valor de la función objetivo *f*, seleccionado uniformemente en el rango de valores posibles (F) $F \in$[Rango]. El individuo de la población con función objetivo *f* más cercano al seleccionado del rango es seleccionado como padre [59].

2.4.1.4 Selección basada en rango

Primero se ordena de forma decreciente a los individuos de una población con base en sus valores de la función objetivo. De esta forma, a cada cromosoma se le otorga un

rango determinado. Los individuos son seleccionados con base en los rangos obtenidos en lugar de sus valores de función objetivo [59].

2.4.1.5 Selección por emparejamiento restringido

Este tipo de selección es utilizada para encontrar padres similares, en donde un padre p1 se empareja con su compañero más similar, éste último se escoge de un grupo de individuos seleccionados aleatoriamente [60].

2.4.2 Cruce

Después de seleccionar los individuos padres, estos son recombinados para producir la descendencia que va a ser parte de la siguiente generación. A continuación se describen las tipos de cruce más conocidos [57, 61].

2.4.2.1 Cruce n puntos

En esta técnica dos cromosomas son cortados en *n* puntos, cada punto de corte o cruce es ubicado en el límite entre dos componentes adyacentes del cromosoma. Luego, el material genético situado entre los *n* puntos es intercambiado. Cada vez que esta técnica es aplicada sobre una pareja distinta de cromosomas, los *n* puntos de corte deben ser seleccionados al azar. Lo más habitual es el uso de cruce de un punto [62] y cruce de dos puntos.

2.4.2.2 Cruce Uniforme

En el cruce uniforme cada gen de la descendencia tiene las mismas probabilidades de pertenecer a uno u otro padre. Se puede implementar de diversas formas, la técnica implica la generación de una máscara de cruce con valores binarios o el cambio de genes de forma aleatoria [63, 64].

2.4.3 Reemplazo

Para mantener el número de individuos de la población, antes de insertar la descendencia en la población se debe realizar un reemplazo. A continuación se muestran los esquemas más reconocidos [65].

2.4.3.1 Reemplazo Peores Individuos

Los individuos que se eliminan de la población para dejar paso a la descendencia se seleccionarán aleatoriamente de un grupo de los peores individuos de la población. Por lo general el grupo está conformado por el 10% de la población [58].

2.4.3.2 Reemplazo por Competencia Restringida

El grupo por competencia restringida está formado con individuos aleatorios de la población actual. En este grupo se busca el peor para ser comparado con el descendiente. Si el valor de la función objetivo del descendiente es mejor que el peor del grupo por competencia restringida, entonces el descendiente reemplaza al peor del grupo [66].

2.4.3.3 Reemplazo Aleatorio

Se seleccionan al azar los individuos que se van a eliminar.

2.4.3.4 Reemplazo de Padres

Consiste en eliminar los padres y adicionar la descendencia.

2.4.3.5 Reemplazo de Individuos Similares

Consiste en reemplazar a un individuo de la población con un ajuste similar al suyo. Para escoger este individuo se obtiene la posición en el que se debe insertar el nuevo individuo y se escoge para agregarlo en una posición al azar.

2.4.4 Búsqueda Local

La búsqueda local busca mejorar la calidad de una descendencia tomando como entrada una solución actual y luego de forma iterativa sustituye la solución actual por otra solución tomada de un vecindario dado. A continuación se describen algunos métodos más conocidos.

2.4.4.1 Búsqueda por vecindad

Consiste en cambiar sistemáticamente de estructura de entornos dentro de la búsqueda para escapar de los mínimos locales [67]. Una estructura de entornos en el espacio de soluciones X es una aplicación N: X → 2^X que asocia a cada solución $x \in X$ un entorno de soluciones $N(x) \subset X$, que se dicen vecinas de *x*. Se tiene un conjunto de vecindarios $(\{N^1, \dots, N^{max}\})$, y una variable *k* que indica el vecindario actual (N^k). Cuando comienza el algoritmo se inicializa $k \leftarrow 1$. Luego, se busca el mejor vecino de *x* en el vecindario (N^1), si la solución encontrada *x'* es mejor que la actual *x*, entonces $x \leftarrow x'$ y se inicializa $k \leftarrow 1$, en caso contrario *k* se incrementa.

2.4.4.2 Búsqueda local Iterada

La idea fundamental es rastrear la solución de un problema combinatorio entre el subespacio definido por los mínimos locales. Un algoritmo de búsqueda transforma una solución cualquiera *s* en *s** que es un óptimo local. Luego para pasar de un óptimo local a otro que sea cercano dentro del subespacio de soluciones, se provoca una perturbación a la solución *s**, lo suficientemente intensa para eludir el óptimo local. Con la perturbación se pasa a otra solución *s'*, a la cual se vuelve a aplicar el algoritmo de búsqueda para alcanzar el nuevo optimo local *s*'*. La metaheurística acepta el paso de *s** a *s*'* mediante algún criterio de aceptación [68].

2.4.4.3 Búsqueda local guiada

Es un método basado en memoria y opera aumentando la función de coste con una penalización: los movimientos de la búsqueda local guiada salen de un mínimo local mediante penalización de determinadas características o elementos de esos mínimos locales que se consideran que deben ocurrir en una solución casi óptima [69].

2.4.4.4 Búsqueda tabú

Es una estrategia que dispone de un mecanismo de memoria, evitando la generación de algunos vecinos dependiendo de la historia reciente, este sistema permite dirigirse a zonas del espacio de soluciones que aún no han sido exploradas [70].

2.4.4.5 Recocido simulado

El recocido simulado es un algoritmo aleatorio de búsqueda local. Las soluciones vecinas de mejor costo son aceptadas e incluso las soluciones de peor costo, aunque con una probabilidad que decrece gradualmente durante el transcurso de la ejecución del algoritmo. La probabilidad de aceptación es controlada por un conjunto de parámetros cuyos valores son determinados por un esquema de enfriamiento [71].

2.4.4.6 Búsqueda Voraz Adaptable Aleatoria

Es un método multiarranque, en el que cada iteración consiste en la construcción de una solución factible aleatoria, y con criterios adaptables para la elección de los elementos a incluir en la solución, seguida de una búsqueda local usando la solución construida como el punto inicial de la búsqueda. Este procedimiento se repite varias veces y la mejor solución encontrada sobre las iteraciones se devuelve como resultado [72].

2.5 ALGORITMO MEMÉTICO

2.5.1 Definición

En 1989, Moscato presentó el término Algoritmo Memético (AM) para describir la combinación de Algoritmos Genéticos con métodos de optimización local [22]. La elección del nombre de estos algoritmos se basa en el concepto de meme definido por Charles Darwin. Los AMs fueron inspirados por modelos de adaptación en sistemas naturales que combinan la adaptación biológica o genética de una población con el aprendizaje que los miembros de esta población pueden lograr durante su tiempo de vida. Los AMs son considerados como una extensión de los Algoritmos Genéticos que aplican técnicas de búsqueda local para mejorar la calidad de las soluciones creadas por la evolución.

Los AMs han demostrado que son más rápidos que los tradicionales algoritmos genéticos para un problema específico, por ejemplo: el problema del vendedor viajero, particionamiento de grafos, planificación de tareas, etc. [73, 74]. En un AM la población se inicializa al azar o mediante una heurística, cada individuo realiza la búsqueda local para mejorar su función objetivo. Para formar una nueva población, son seleccionados los individuos de alta calidad [75].

Los pasos principales de un AM, incluyen generar la población inicial y evaluar la función objetivo de cada agente de ésta, seleccionar los padres que se van a cruzar para generar los hijos, optimizar los hijos por medio de búsqueda local y actualizar la población actual con los descendientes por medio del reemplazo [76].

Capítulo 3

3 ENTORNO HIPERHEURÍSTICO

3.1 HIPERHEURÍSTICA CONSTRUCTIVA CON APRENDIZAJE

En este proyecto se propone una hiperheurística constructiva con aprendizaje en línea basada en el esquema de la Figura 1 para obtener el algoritmo memético. Es un aprendizaje en línea porque se guardan los valores de las probabilidades de los esquemas de bajo nivel durante la ejecución de la hiperheurística; estos valores permiten realizar la selección de un esquema teniendo en cuenta su desempeño.

En la Figura 2 se puede observar el esquema de la hiperheurística propuesta, que está compuesto por el controlador de dominio y el dominio del problema, de la siguiente forma:

- Medidas de desempeño. Obtiene los contadores de los esquemas de bajo nivel (selección, cruce y búsqueda local) y calcula sus probabilidades con la Ecuación 8 de la sección 3.3.1$W_{t,s} = TF_{t,s} \times ISF_t$.

- Controlador de la hiperheurística. Utiliza los esquemas de selección de alto nivel ruleta y torneo probabilístico explicados en la sección 3.3.1 para elegir los esquemas de bajo nivel teniendo en cuenta las probabilidades que tienen cada uno de ellos.

- Conjunto de heurísticas de bajo nivel. Está conformado por las heurísticas de selección, cruce y búsqueda local que son utilizadas en la configuración del algoritmo memético para generación automática de resúmenes de múltiples documentos.

- Soluciones. Utiliza la configuración de los esquemas de bajo nivel en el algoritmo memético.

- Función objetivo. Evalúa la calidad de la solución obtenida con los esquemas de bajo nivel, si la solución con esta configuración es factible entonces los contadores de estos esquemas se incrementan en caso contrario se disminuyen.

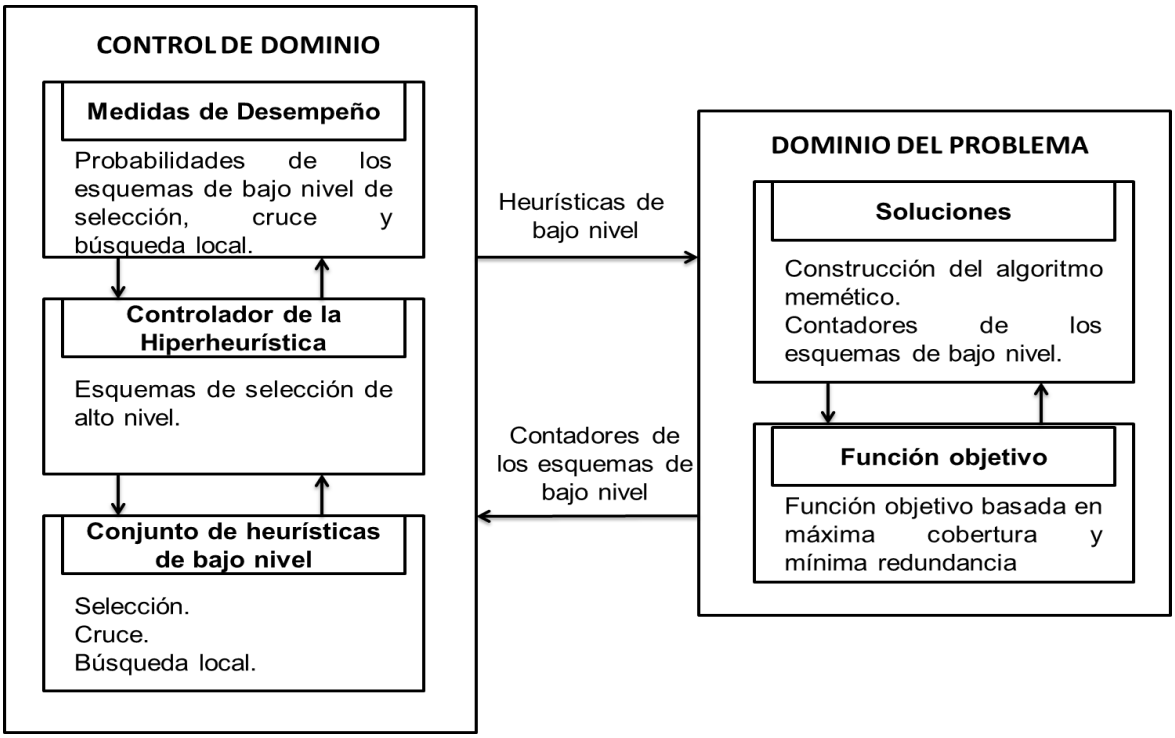

Figura 2. Esquema de la hiperheurística propuesta

3.1.1 Esquemas utilizados en la hiperheurística

Para escoger los esquemas de alto y bajo nivel a utilizar en el entorno hiperheurístico, se realizó una revisión de los esquemas más conocidos de selección, cruce, reemplazo y búsqueda local, teniendo en cuenta como criterios la calidad y diversidad de las soluciones (Ver Anexo A).

3.1.1.1 Esquemas de alto nivel

Para escoger los esquemas de alto nivel se tuvo en cuenta que estos eligen heurísticas de bajo nivel para ser utilizados en la resolución del problema de generación automática de resúmenes para múltiples documentos. Por lo tanto, se escogió la selección por ruleta ya que permite elegir las mejores heurísticas para resolver el problema, además ha sido utilizado como heurística de alto nivel en la hiperheurística constructiva de reducción de atributos obteniendo soluciones de buena calidad [20], y la selección por torneo probabilístico para elegir las mejores y peores heurísticas que ayudan a resolver el problema planteado obteniendo diversidad en las heurísticas, de esta manera explorar nuevas soluciones del espacio de búsqueda.

3.1.1.2 Esquemas de bajo nivel

Para los esquemas de selección de bajo nivel se utilizaron: la selección por ruleta porque obtiene soluciones de buena calidad y es sencilla de implementar; y la selección por emparejamiento restringido porque fomenta la creación de nuevas generaciones manteniendo la diversidad de la población. A pesar, que la selección por torneo probabilístico tiene en cuenta la diversidad de la población, se utilizó la selección por emparejamiento restringido porque además de permitir variedad en la población al

explorar nuevas soluciones del espacio de búsqueda eligiendo el primer padre de forma aleatoria, se va a mantener la descendencia con gran parte de los memes de sus padres debido a que los padres son similares.

Con respecto a los esquemas de cruce se tuvo en cuenta un esquema clásico y sencillo como el cruce Unipunto, que ha sido utilizado en diversos proyectos [62, 77, 78]; y el cruce uniforme que permite variedad en los memes de un agente, ya que selecciona aleatoriamente los memes de cada padre, para crear los descendientes, de esta manera poder obtener descendientes lo suficientemente diferentes de sus padres con el fin de explorar el espacio de búsqueda.

Se utilizaron el reemplazo de los peores individuos porque permite tener soluciones de buena calidad reduciendo las soluciones de baja calidad y el reemplazo por competencia restringida porque los nuevos individuos tienen mayor probabilidad de reemplazar individuos de la población similares a ellos según su genotipo [66], de esta manera en la población no se tiene un exceso de soluciones similares, que podrían llevar a la convergencia prematura.

Por último, se utilizaron búsqueda por vecindad y búsqueda local iterada, teniendo en cuenta que son sencillas de implementar y han obtenido buenos resultados en diferentes problemas de optimización tales como: aprendizaje en redes bayesianas, clasificación y planificación; asignación cuadrática (QAP), particionamiento de grafos, máxima satisfacción (MAX-SAT) y en el problema del árbol de Steiner restringido a grafos [68, 79]. La búsqueda por vecindad evita quedarse en mínimos locales, escapa de ellos cambiando de forma sistemática la estructura de entornos, y la búsqueda local iterada escapa de los mínimos locales utilizando la perturbación. A pesar de que la búsqueda tabú y local guiada, obtienen buenos resultados en problemas de optimización éstos requieren la afinación de parámetros como el ajuste de la lista tabú, la elección de los movimientos que se deben registrar y la definición del criterio de aspiración en la búsqueda tabú; para la búsqueda local guiada el ajuste de la penalización y la regularización.

3.2 PROCESO DEL ENFOQUE HIPERHEURÍSTICO PARA OBTENER EL ALGORITMO MEMÉTICO

El proceso del enfoque hiperheurístico para obtener el algoritmo memético teniendo en cuenta los pasos de la Figura 2 es descrito a continuación:

Control de dominio

En este trabajo se realizaron dos experimentaciones con las selecciones de alto nivel: una con la selección de ruleta y la otra con la selección de torneo probabilístico descritos en la sección 3.3.1. Estos fueron los encargados de seleccionar las heurísticas de bajo nivel selección, cruce y búsqueda local teniendo en cuenta los siguientes pasos:

1. Se crea una lista de probabilidades de los esquemas de bajo nivel. De acuerdo a los valores de la siguiente Tabla 1.

Posición	Esquema	Esquemas de bajo nivel	Probabilidades
0	Selección	Selección por Ruleta (SR)	0.5
1		Selección por Emparejamiento Restringido (SER)	0.5
2	Cruce	Cruce Unipunto (Cup)	0.5
3		Cruce Uniforme (Cun)	0.5
4	Búsqueda Local	Búsqueda por vecindad greedy con distancia de hamming 1 y 2 (VNDDH1YDH2Greedy).	0.166666
5		Búsqueda por vecindad aleatorio con distancia de hamming 1 y 2 (VNDDH1YDH2Aleatorio).	0.166666
6		Búsqueda local iterada por vecindad greedy distancia de hamming 1 y 2 (ILDH1yDH2Greedy).	0.166666
7		Búsqueda local iterada por vecindad aleatoria distancia de hamming 1 y 2 (ILDH1y DH2Aleatoria).	0.166666
8		Búsqueda local iterada por vecindad greedy distancia de hamming 1 (ILDH1Aleatorio).	0.166666
9		Búsqueda local iterada por vecindad aleatoria distancia de hamming 2 (ILDH2Aleatorio).	0.166666

Tabla 1. Inicialización de las probabilidades de los esquemas de bajo nivel

La lista de valores iniciales de los esquemas de bajo nivel se puede observar en la Figura 3.

Probabilidades	0.5	0.5	0.5	0.5	0.16	0.16	0.16	0.16	0.16	0.16
Posición	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

Figura 3. Lista de valores iniciales de los esquemas de bajo nivel

2. Una heurística de alto nivel elige una heurística de bajo nivel teniendo en cuenta las probabilidades en cada conjunto (selección, cruce y búsqueda local) como se explica en la sección 3.3.1. Se debe tener en cuenta que primero se utilizó un esquema de alto nivel y luego el otro.

Para el conjunto de heurísticas de bajo nivel se utilizaron: selección por ruleta y por emparejamiento restringido; cruce unipunto y uniforme; reemplazo por competencia restringida y de los peores individuos; búsqueda por vecindad variable y local iterada; descritas en las secciones 3.3.2.1, 3.3.2.2, 3.3.2.3 y 3.3.2.4 respectivamente.

Al principio todas las heurísticas de bajo nivel tienen la misma probabilidad de ser seleccionadas, pero durante la ejecución estas probabilidades cambian, de acuerdo a los resultados que se obtienen con estos esquemas de bajo nivel.

Dominio del problema

Con estos esquemas seleccionados para la solución actual se construye un algoritmo memético (conformado de un esquema de selección, cruce y búsqueda local), éste internamente obtiene una solución y la evalúa, si el valor obtenido de la función objetivo en la evaluación es bueno, entonces se incrementa el contador en cada uno de estos esquemas elegidos, en caso contrario el contador va disminuyendo; estos contadores son usados para calcular las probabilidades. Estas probabilidades permiten a los esquemas de alto nivel tenerlas en cuenta para ser elegidas o no.

En la Figura 4 se puede observar el funcionamiento de la hiperheurística propuesta, inicialmente se elige una heurística de alto nivel: selección por ruleta o selección por torneo probabilístico, luego con esta heurística se procede a elegir un esquema de bajo nivel de selección, uno de cruce y uno de búsqueda local. Con estos esquemas se construye el algoritmo memético, junto con un esquema de reemplazo que esta fijo en el algoritmo. Esta configuración del algoritmo memético obtiene una solución que es evaluada con la función objetivo, si esta solución da buenos resultados los contadores para estos esquemas de bajo nivel se incrementan en caso contrario se disminuyen. Como se mencionó anteriormente con estos contadores se calcula la probabilidad de Laplace y con base en esta probabilidad se escogen los nuevos esquemas de bajo nivel. Estos pasos se repiten hasta que termine un número máximo de evaluaciones.

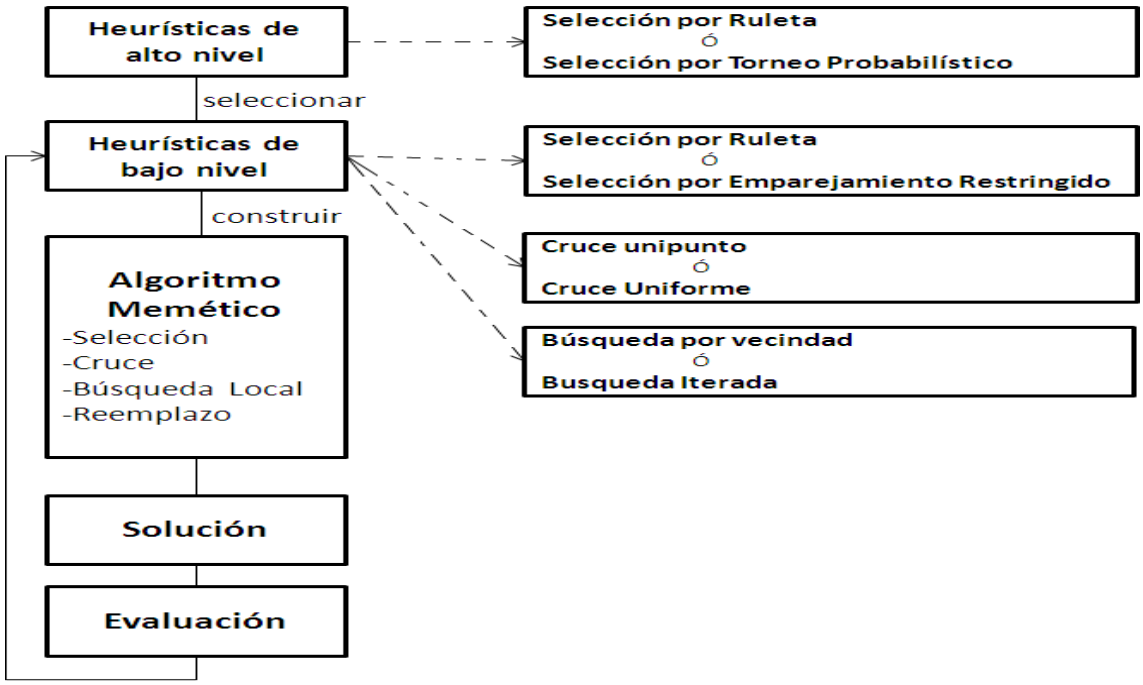

Figura 4. Funcionamiento de la hiperheurística

3.3 ESQUEMAS DE SELECCIÓN DE ALTO NIVEL Y ESQUEMAS DE BAJO NIVEL (selección, cruce, reemplazo y búsqueda local)

3.3.1 Esquemas de selección de alto nivel

Se encarga de elegir una heurística de bajo nivel (selección, cruce y búsqueda local) teniendo en cuenta la probabilidad que tiene asignada. Sea *n* la cantidad de esquemas de un mismo grupo (ejemplo: selección), $N_{éxitos}$ la cantidad de veces que ha sido seleccionado el esquema, $T_{éxitos}$ el total de éxitos de los esquemas (selección, cruce o búsqueda local). La probabilidad asociada a su selección está dada por la Ecuación 8 que define la probabilidad de Laplace [80].

$$P = (1 + N_{éxitos})/(T_{éxitos} + n) \qquad \text{Ecuación 8}$$

3.3.1.1 Selección por Ruleta.

Esta selección permite que los mejores esquemas de bajo nivel sean elegidos con una mayor probabilidad. En la Figura 5 se pueden ver los pasos a seguir para la selección por ruleta.

01 Calcular la suma total de las probabilidades en cada conjunto de los esquemas.
02 Repetir N veces (N es el tamaño del conjunto).
03 Generar un número aleatorio *r* entre 0 y 1.
04 Recorrer secuencialmente las posiciones de los esquemas, sumando los valores esperados, hasta que la suma sea mayor o igual a *r.*
05 La posición del esquema que haga que la suma exceda el límite *r* es el seleccionado.

Figura 5. Selección por Ruleta de Alto Nivel

3.3.1.2 Selección por Torneo Probabilístico

Al igual que la selección por ruleta elige una heurística de bajo nivel de cada conjunto, los pasos se muestran en Figura 6. Esta selección tiene en cuenta a los mejores y peores esquemas para ser elegidos.

01 Escoger un número *p* esquemas (generalmente 2).
02 Compararlos con base en su probabilidad.
03 Para el ganador del "torneo" se genera un número aleatorio del intervalo [0...1], si es mayor que un parámetro p (p = 0.5) se escoge el individuo más alto y en caso contrario el menos apto.

Figura 6. Selección por Torneo Probabilístico de Alto Nivel

3.3.2 Esquemas de bajo nivel

A continuación se describen los esquemas utilizados en el enfoque hiperheurístico para la construcción del algoritmo memético.

3.3.2.1 Selección

El proceso de selección se encarga de elegir parejas de agentes[9] de la población actual que serán padres de los nuevos agentes, los cuales podrán formar parte de la nueva población. Para la selección de estos esquemas se tuvo en cuenta la calidad por medio de la selección por ruleta que casi siempre elige los mejores agentes [81]; y la diversidad con la selección por emparejamiento restringido, que tiene en cuenta los mejores y peores agentes [60].

3.3.2.1.1 Selección por ruleta

Cada uno de los agentes de la población tiene un valor de la función objetivo proporcional a su calidad, de tal forma que la suma de estos agentes sea el total de la función objetivo de la población. Los mejores agentes recibirán una porción de la ruleta mayor que la recibida por los peores. Generalmente la población está ordenada con base en el mejor valor de la función objetivo por lo que las porciones más grandes se encuentran al inicio de la ruleta. Para seleccionar un agente basta con generar un número aleatorio del intervalo [0..1] y devolver el agente situado en esa posición de la ruleta. Esta posición se suele obtener recorriendo los agentes de la población y acumulando sus proporciones de ruleta hasta que la suma exceda el valor obtenido. Para esta selección se manejan los mismos pasos de la Figura 5, pero en el paso uno se calcula la suma de la función objetivo de la población; en dos, N es el tamaño de la población; y los pasos cuatro y cinco se manejan agentes de la población en lugar de esquemas.

3.3.2.1.2 Selección por Emparejamiento Restringido

Se escoge una solución similar a la solución inicial que se encuentre en el grupo de selección [82]. En la Figura 7 se muestran los pasos que se realizaron para este esquema.

Pasos:
01 El *padre1* (*P1*), se escoge aleatoriamente de la población actual.
02 Se genera un grupo de tamaño pequeño, escogido aleatoriamente de la población actual.
03 El *padre2* (*P2*), se escoge del grupo de selección teniendo en cuenta que sea similar a *P1*. Se utiliza la similitud de cosenos para escoger el *P2*.

Figura 7. Pasos de la selección por Emparejamiento Restringido

3.3.2.2 Cruce

Es el intercambio de material genético entre dos agentes con el objetivo de dar origen a nuevos agentes [57].

[9] Agente. es un vector con ceros y unos. Las posiciones del agente que tienen uno significa que la frase es tenida en cuenta para el resumen en caso contrario no lo es.

3.3.2.2.1 Cruce Unipunto

En este esquema se elige un punto intermedio aleatoriamente en los dos agentes padres, dividiendo los padres en ese punto de cruce, y creando los hijos mediante el intercambio de las colas de los agentes, como se puede observar en la Figura 8.

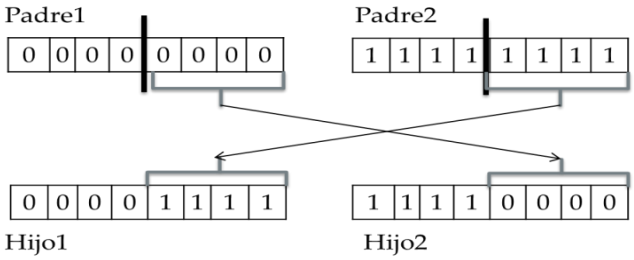

Figura 8. Cruce Unipunto

3.3.2.2.2 Cruce Uniforme

Dados dos agentes denominados Padre 1 y Padre 2, en este esquema el i-ésimo meme[10] del primer agente hijo se elige al azar entre el Padre 1 y el Padre 2. Si el Padre 1 es seleccionado, entonces el i-ésimo meme del primer hijo será igual al i-esimo meme del Padre 1. Además, el i-ésimo meme del segundo hijo será igual al i-ésimo meme del Padre 2. Si el Padre 2 es seleccionado, la asignación de memes será inversa a la mencionada. Ver Figura 9.

Figura 9.Cruce Uniforme

3.3.2.3 Reemplazo

El reemplazo o actualización se encarga de limitar el tamaño de la población, esto es eliminar algunos agentes para permitir la entrada de otros nuevos. Esta técnica se manejó de forma fija, debido a que el reemplazo tan sólo define el agente que queda en la nueva población.

3.3.2.3.1 Reemplazo por Competencia Restringida

En este esquema se forma un grupo de agentes aleatorios de la población actual, en este grupo se busca el peor agente para ser comparado con el descendiente. Si la función de

[10] Meme, es un valor 0 u 1 que están distribuidos en el agente.

aptitud del descendiente es mejor que el peor del grupo por competencia restringida, entonces el descendiente reemplaza al peor del grupo como se observa en la Figura 10, en este caso el descendiente es mejor y reemplaza al agente 5 de la población actual.

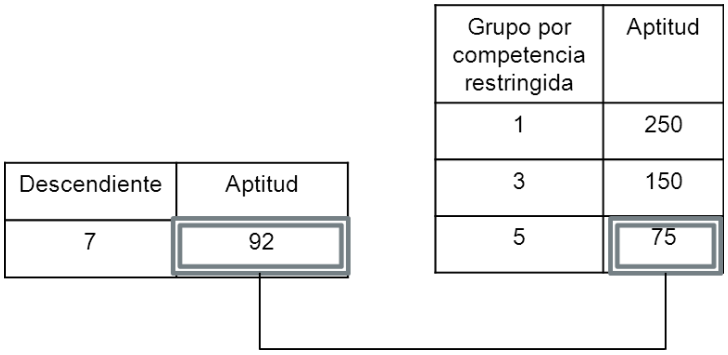

Figura 10. Ejemplo de Reemplazo por Competencia Restringida

3.3.2.3.2 Reemplazo de los Peores Individuos

En este reemplazo se elige un porcentaje aleatorio de los peores agentes de la población, los cuales serán reemplazados por los descendientes. En la Figura 11 se observa un ejemplo de este tipo de reemplazo.

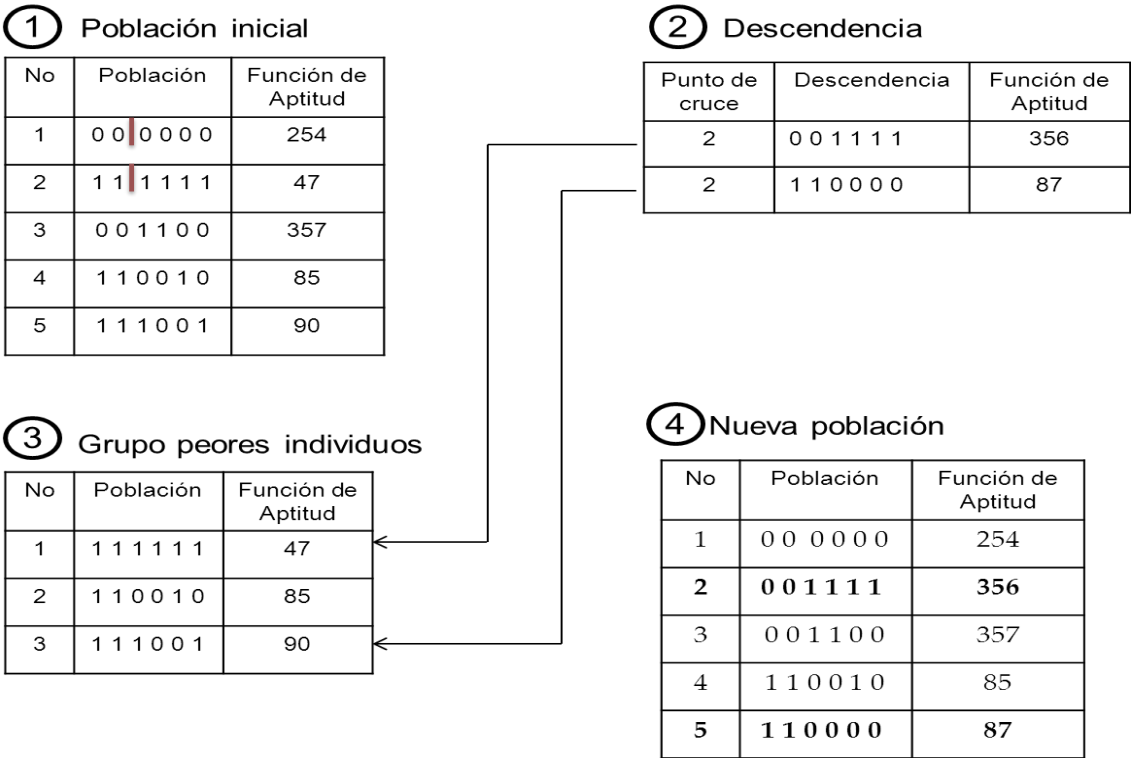

Figura 11. Ejemplo del Reemplazo de los Peores Individuos

3.3.2.4 Búsqueda local

3.3.2.4.1 Búsqueda por vecindad

La búsqueda de Entorno Variable (VNS, por sus siglas en inglés Variable Neighbourhood Search) es una metaheurística para resolver problemas de optimización cuya idea básica es el cambio sistemático de entorno dentro de una búsqueda local [67].

Los vecinos se generaron teniendo en cuenta la misma distancia de hamming[11] entre la solución actual y la solución optimizada. Se definieron dos vecindarios: con distancia de hamming uno, que cambia el valor de un meme en el agente actual; y con distancia de hamming dos, que modifica dos memes del agente actual. El cambio consiste en cambiar de uno a cero, o viceversa.

El vecindario con distancia de hamming uno greedy, es un conjunto de aquellas soluciones que a partir del agente inicial actual se le adiciona un meme con la cobertura más alta. Los memes se encuentran en una lista ordenados descendentemente por cobertura. A medida que se van generando más vecinos se adiciona el meme siguiente de la lista, siempre y cuando no se encuentre en la solución inicial actual.

El vecindario con distancia de hamming dos, es un conjunto de soluciones que a partir del agente inicial actual se elimina un meme con peor cobertura y se adiciona un meme con la cobertura más alta. Los memes también se manejan con la lista ordenada por cobertura.

Se utilizaron dos búsquedas locales de vecindad:

- Búsqueda por vecindad greedy con distancia de hamming 1 y 2 (VNDDH1YDH2Greedy): encuentra una mejor solución a partir de los vecinos de la solución inicial actual (Ver Figura 12). La mejor solución es aquella que tenga la mayor función objetivo con el conjunto de oraciones de la colección de documentos. Cambiando a la estructura de vecinos si no se obtiene mejora y volviendo a la estructura de vecinos en otro caso.

[11] Distancia de hamming, consiste en el número de bits que tienen que cambiarse para transformar una palabra de código válida en otra palabra de código válida

```
01    Solución actual.
02    Repetir hasta encontrar mejor solución.
02.1    Vecindario con distancia de hamming uno.
02.2    Mientras vecindario con distancia de hamming menor que tres.
02.2.1    Optimización por vecindad greedy con distancia de hamming actual (uno o
          dos).
0.2.2.2    Si solución optimizada es mejor que la actual
0.2.2.2.1    Volver al paso 02.1.
0.2.2.3    Sino.
0.2.2.3.1    Volver al paso 02.2 teniendo en cuenta el vecindario con distancia de
             hamming dos.
0.2.3    Fin mientras.
0.3    Fin repetir.
```

Figura 12. Búsqueda por vecindad greedy con distancia de hamming 1 y 2 (VNDDH1YDH2Greedy)

- Búsqueda por vecindad aleatoria con distancia de hamming 1 y 2 (VNDDH1YDH2Aleatorio): encuentra una solución aleatoria a partir de los vecinos de la solución inicial actual, cambiando a la estructura de vecinos si no se obtiene mejora y volviendo a la primera estructura de vecinos en otro caso. Se utilizan los mismos pasos de la Figura 12, el único cambio que presenta es el paso 03 dado que se realiza es la optimización por vecindad aleatoria.

3.3.2.4.2 Búsqueda Local Iterada

Es una metaheurística que usa una solución inicial, una búsqueda local y un procedimiento de perturbación. La perturbación consiste en realizar un cambio o alteración a la solución actual [68].

Se utilizaron cuatro algoritmos de búsqueda para la optimización de los agentes (búsqueda por vecindad greedy con distancia de hamming uno y con distancia de hamming dos, búsqueda por vecindad aleatorio con distancia de hamming uno y con distancia de hamming dos). Los pasos de la búsqueda local iterada son descritos en la Figura 13. En el punto uno no solo se tiene en cuenta la primera solución inicial, debido a que este es un problema poblacional donde se obtienen diferentes soluciones en cada iteración. En el punto 3, la perturbación se realiza si la solución se encuentra en la nueva población. Para este problema la perturbación consiste en generar un agente aleatorio.

```
01  Solución actual.
02  Aplicar un algoritmo de búsqueda que proporcione un óptimo local s*.
03  Si solución optimizada s* está en la nueva población.
03.1  Aplicar una perturbación a la solución s* para transformarla en s'.
03.2  Emplear el algoritmo de búsqueda para obtener s*'.
```

Figura 13. Búsqueda local iterada

Capítulo 4

4 ALGORITMO MEMÉTICO PARA GENERACIÓN DE RESÚMENES DE MÚLTIPLES DOCUMENTOS

4.1 ESQUEMA GENERAL DEL SISTEMA PROPUESTO DE GENERACIÓN AUTOMÁTICA DE RESÚMENES

En la Figura 14 se muestra el esquema general del sistema de generación de resúmenes. Inicialmente se procesan los documentos realizando la segmentación de oraciones (unas quedan almacenadas en una lista para ser utilizadas en el resumen y estas mismas oraciones quedan almacenadas en otra lista para ser procesadas), filtro de palabras vacías y lematización para reducir las palabras poco significativas. De esta forma poder realizar la representación de los documentos cuyo objetivo es facilitar la interpretación de los documentos para que puedan ser procesados por el algoritmo memético. El algoritmo memético es el encargado de seleccionar el agente resumen con las oraciones más relevantes teniendo en cuenta la función objetivo basada en cobertura y relevancia cuyo resultado es el conjunto de oraciones que conforman el resumen.

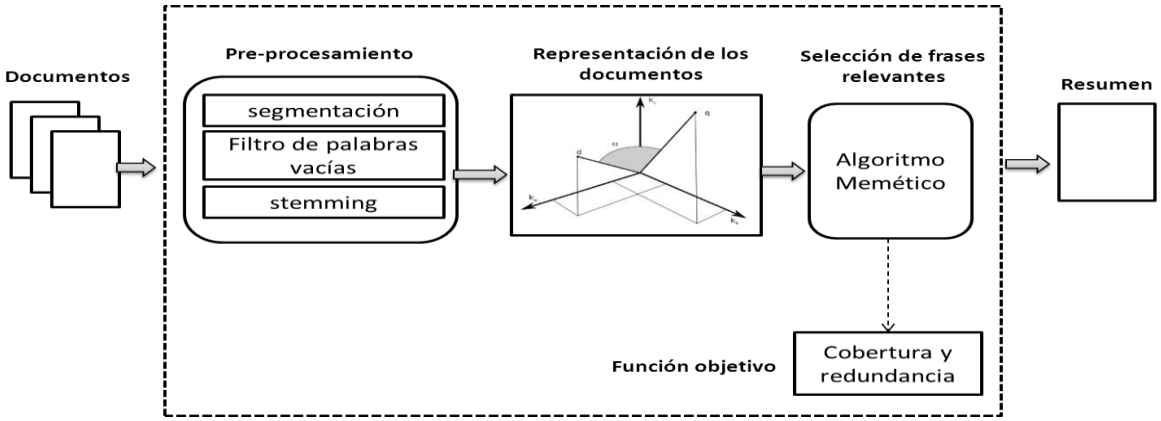

Figura 14. Esquema del sistema propuesto de generación automática de resúmenes.

4.2 FUNCIÓN OBJETIVO

La selección de la función objetivo es muy importante en el algoritmo memético para generación de resúmenes, debido a que de ella depende la puntuación que se le da a los resúmenes candidatos. En este proyecto se plantea el uso de una función objetivo basada en máxima cobertura y mínima redundancia, teniendo en cuenta que investigaciones que contemplan estos factores en la función objetivo han mostrado buenos resultados con respecto al estado del arte [19, 83-85]. Una de estas investigaciones es el algoritmo basado en PSO cuya función objetivo está compuesta por estos dos factores (MCMR-PSO) [19], y que fue tomada como base para la función objetivo del algoritmo propuesto en este proyecto.

La función objetivo es definida como la combinación lineal de los factores de cobertura (FC) y redundancia (FR) como se plantea en la Ecuación 9, los cuales están controlados por el coeficiente lambda (λ) el cual le da flexibilidad a la función objetivo permitiendo que se dé mayor o menor peso a cada uno de los factores. El coeficiente λ varía entre 0 y 1.

Ecuación 9

$$f(x) = \lambda * FC - (1 - \lambda) * FR$$

4.2.1 Factor de Cobertura (FC)

En este trabajo, la cobertura busca seleccionar las oraciones más relevantes con respecto a las oraciones de los documentos, para lo cual es importante medir la similitud entre el texto del resumen (todas las oraciones candidatas del resumen) y las oraciones de toda la colección de documentos. De esta forma el factor se calcula como se muestra en la Ecuación 10.

$$FC = Sim\ (R, D) \qquad \text{Ecuación 10}$$

Donde *R*, representa el texto con todas las oraciones del resumen de la solución candidata; *D*, representa todas las oraciones de la colección de documentos (en este caso es el centroide de la colección); y *Sim*(*R*,*D*), es la similitud de cosenos entre el vector de términos de *R* y el vector de términos de *D* (calculada con la Ecuación 7). Por lo tanto este factor toma valores entre cero y uno.

4.2.2 Factor de Redundancia (FR)

Un resumen sin redundancia es aquel que contiene oraciones que no expresan la misma información, por el contrario, un resumen es redundante si las oraciones que están en el resumen tratan sobre el mismo tema. Este factor de redundancia se calcula teniendo en cuenta la similitud promedio de las oraciones. Es deseable que el valor de este factor sea pequeño, porque quiere decir que las oraciones del resumen son distintas; si el valor es alto indicaría que las oraciones del resumen se parecen entre sí. Este factor de redundancia fue tomado de la misma forma como se planteó en MCMR-PSO, pero normalizando el valor que tome valores entre cero y uno, al igual que el factor de cobertura (Ver Ecuación 11).

$$FR = \frac{2}{n \times (n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Sim(S_i, S_j) \qquad \text{Ecuación 11}$$

Donde S_i y S_j son oraciones del resumen $Sim(S_i, S_j)$ es la similitud de cosenos entre las dos oraciones (calculada con la Ecuación 7) y *n* es la cantidad de oraciones que hay el resumen.

4.3 REPRESENTACION DEL AGENTE

El agente representa las oraciones candidatas que podrían ser parte del resumen (ver Figura 15). La longitud del agente corresponde a la cantidad de oraciones que componen la colección de documentos, que se obtuvo con un segmentador. La posición 1 del agente representa la oración 1, la posición 2 la oración 2, …, la posición n la oración n. cada posición del agente está representado de forma binaria (0,1) donde uno indica que la oración se toma en cuenta para el resumen y cero que la oración es descartada.

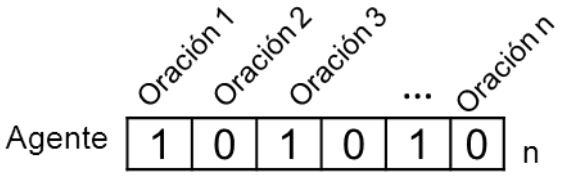

Figura 15. Representación vectorial del agente

4.4 ESQUEMA GENERAL DEL ALGORITMO MEMÉTICO

En la Figura 16 se presenta el esquema general del algoritmo memético (AM) obtenido desde el enfoque hiperheurístico, con base en el esquema presentado en [76]. Como se puede apreciar los pasos principales de un AM incluyen generar la población inicial y evaluar la función objetivo de cada agente de esta, seleccionar los padres que se van a cruzar para generar los hijos, optimizar los hijos por medio de búsqueda local y actualizar la población actual por los descendientes por medio del reemplazo.

Entrada: Documentos DUC.
Salida: Resumen.

01 Población Inicial
02 Evaluación de la función objetivo (basada en cobertura y redundancia) de cada agente de la población.
03 Mientras Máximo número de evaluaciones de la función objetivo no se cumpla hacer
03.1 Selección por emparejamiento restringido de dos padres
03.2 Cruce unipunto, dos hijos son generados
03.3 Mejorar los descendientes con la búsqueda por vecindad greedy con distancia de haming 1 y 2.
03.4 Actualizar la población con el reemplazo por peores individuos.

Figura 16. Esquema general del Algoritmo Memético

- **Entrada**: conjunto de documentos DUC cuya estructura se muestra en la Figura 28, de la sección 5.2. Estos documentos se representaron en el espacio vectorial como se explicó en la sección 2.2.

- **Salida:** se obtiene el agente resumen que tiene mejor valor en su función objetivo. El resumen generado se obtiene concatenando las oraciones que presentan un uno en el agente. Estas oraciones en su forma original se encuentran almacenadas en la lista de oraciones que se obtuvo con el algoritmo de segmentación que será explicado en la sección 5.1.1. En fin, el resumen está formado por las oraciones más representativas del texto original, a las que no se les realizó un estudio de coherencia y legibilidad.

- **Población inicial (01):** Para el caso de generación de resúmenes se consideran cincuenta agentes (teniendo en cuenta el tamaño de la población del método MCMR-PSO) de longitud *n*, donde *n* es el número de oraciones candidatas (las oraciones

candidatas elegidas fueron las que tuvieron una similitud con el documento mayor a un umbral de 0.13, eliminando las oraciones que no superaron este umbral tal como se explica en la sección 5.1.4). En la Figura 17 se puede observar un en ejemplo de la población inicial.

- **Inicialización del agente resumen**
  Cada agente de la población es inicializado de forma aleatoria, seleccionando oraciones hasta completar la longitud del resumen que no supere un máximo de 250 palabras. Las oraciones seleccionadas para el agente resumen se activan y las que no se desactivan.

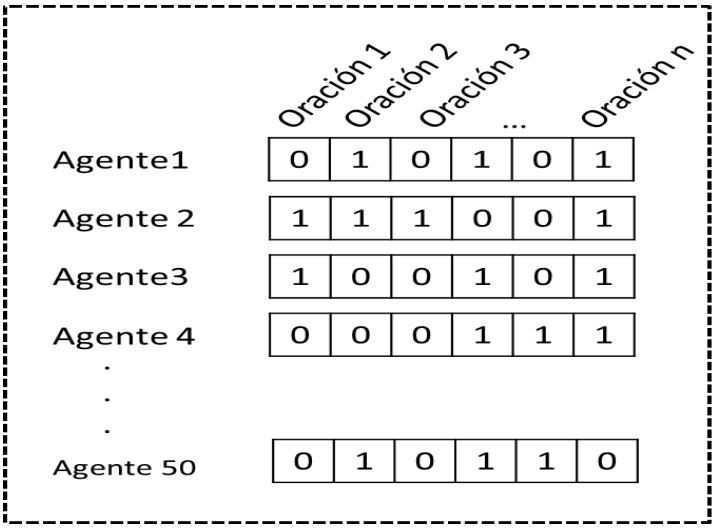

Figura 17. Población inicial

- **Evaluación de la función objetivo (02):** cada uno de los agentes de la población es evaluado con la función objetivo explicado en la sección 4.2. A cada agente se calcula:
  - El factor de cobertura (Ver Ecuación 10): calcula la similitud de cosenos entre el vector de pesos de términos de las oraciones candidatas y el vector de pesos de términos de las oraciones del conjunto de documentos.
  - Y el factor de redundancia (ver Ecuación 11): calcula la similitud de cosenos entre el vector de pesos de términos de las oraciones candidatas para el resumen, uno a uno.
    Estos dos factores se restan para obtener el valor de la función objetivo para el agente resumen, teniendo en cuenta un coeficiente lambda que equilibra los pesos de ambos factores. Esta función objetivo se realiza con el fin de evaluar la calidad de cada uno de los agentes resumen de la población.

- **Selección de padres (03.1):** determina los agentes candidatos que son usados para crear nuevos agentes. Se realiza con la selección por emparejamiento restringido descrito en la sección 3.3.2.1.2.

- **Cruce (03.2):** crea dos nuevos agentes candidatos por medio del cruce unipunto entre los padres, descrito en la sección 3.3.2.2.1.

- **Búsqueda local (03.3):** el objetivo es mejorar la calidad de la descendencia con la búsqueda por vecindad greedy con distancia de hamming 1 y 2 descrita en la sección 3.3.2.4.1.

- **Actualizar la población (03.4):** este paso determina si un nuevo agente debe ser parte de la nueva población y cuál es el agente que debe ser reemplazado, esta decisión se toma con el reemplazo de los peores individuos explicado en la sección 3.3.2.3.2.

- **Condición de parada (03):** La ejecución del algoritmo termina cuando se cumple con el número máximo de evaluaciones de la función objetivo (15000), de lo contrario repetir los pasos 03.1, 03.2, 03.3 y 03.4.

4.5 COMPORTAMIENTO DEL ALGORITMO MEMÉTICO

En la Figura 18 se muestra el comportamiento del algoritmo memético para cada una de las seis generaciones (la cantidad de generaciones aumentaba teniendo en cuenta que no se superará el máximo número de evaluaciones de la función objetivo que para este proyecto son 15000), con un conjunto de documentos de DUC2005. El algoritmo memético se ejecutó treinta veces y por cada generación se obtenía el valor de la función objetivo del mejor agente resumen. Al final se calculó el promedio de los valores de la función objetivo de cada generación obtenida en cada ejecución.

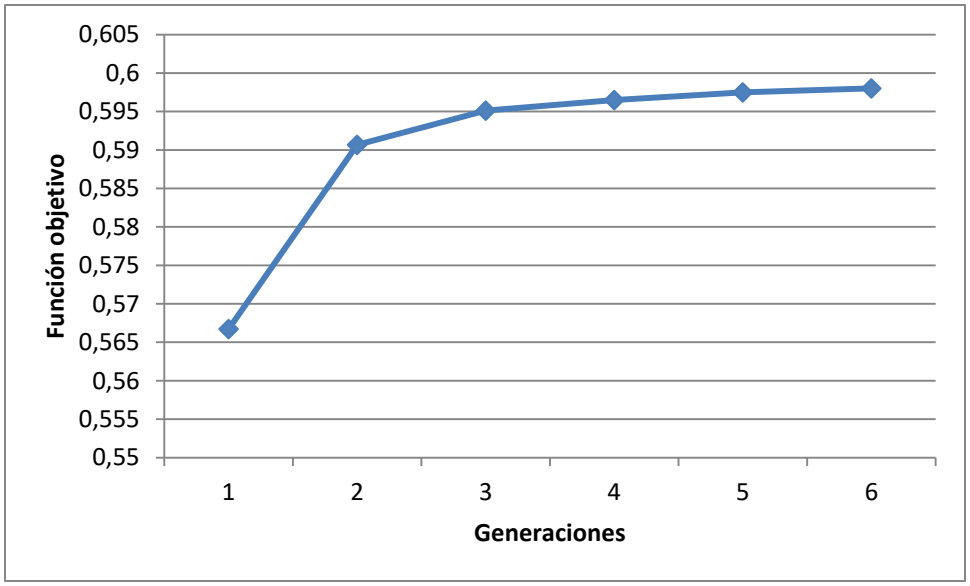

Figura 18. Comportamiento del algoritmo memético

4.6 AFINACIÓN DE PARÁMETROS

Los parámetros del algoritmo memético obtenido desde el enfoque hiperheurístico que fueron afinados son los siguientes:

- La probabilidad de optimización (PO), este parámetro tomó los valores de {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1}.

- El tamaño de la población (TP), para este valor se partió de un tamaño de 30 con incrementos de 10 hasta llegar a 100.

- Lambda de la función objetivo (LF), se partió de 0.7 con incrementos de 0.2 hasta llegar a 0.98.

- Máxima longitud del resumen (MLR), para este valor se partió de 270 con incrementos de 20 hasta llegar a 390.

La afinación consistió en tomar cada parámetro del algoritmo memético, buscar el mejor valor de afinación con el primer parámetro, luego con el siguiente, así hasta terminar con los parámetros a afinar.

Esta afinación se realizó tanto para el conjunto de datos de DUC2005, como para DUC2007, teniendo en cuenta el manejo de la mejor configuración hiperheurística obtenida. El cálculo de la función objetivo para un conjunto de datos consistió en ejecutar 30 veces el algoritmo y se promedió la medida de ROUGE obtenida en cada experimento.

4.6.1 Resultados de la afinación

En la Tabla 2 se muestran los resultados de la afinación. Los valores de afinación de parámetros en el algoritmo memético fueron diferentes en cada conjunto de documentos.

Parámetros	**DUC2005**	**DUC2007**
Probabilidad de Optimización	0.5	0.4
Tamaño de la Población	70	50
Lambda	0.86	0.86
Máxima Longitud del Resumen	290	270

Tabla 2. Afinación de parámetros

Los valores de los parámetros que se dejaron para el algoritmo memético son los de DUC2007, esto se obtuvo por medio de varias experimentaciones como se puede observar en las siguientes figuras de las secciones 4.6.1.1,4.6.1.2 y 4.6.1.3 (para mayor detalle ver Anexo F).

4.6.1.1 Afinación para DUC2005

En la Figura 19 se puede observar que se obtuvieron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con un valor de probabilidad de optimización de 0.5.

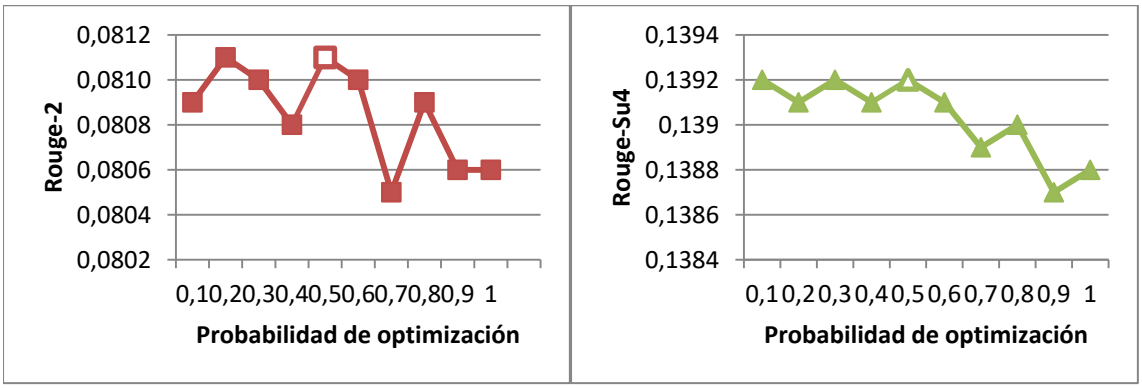

Figura 19. Afinación de la probabilidad de optimización

En la Figura 20 utilizando el mejor valor afinado de la probabilidad de optimización se puede observar que se lograron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con un tamaño de población de 70.

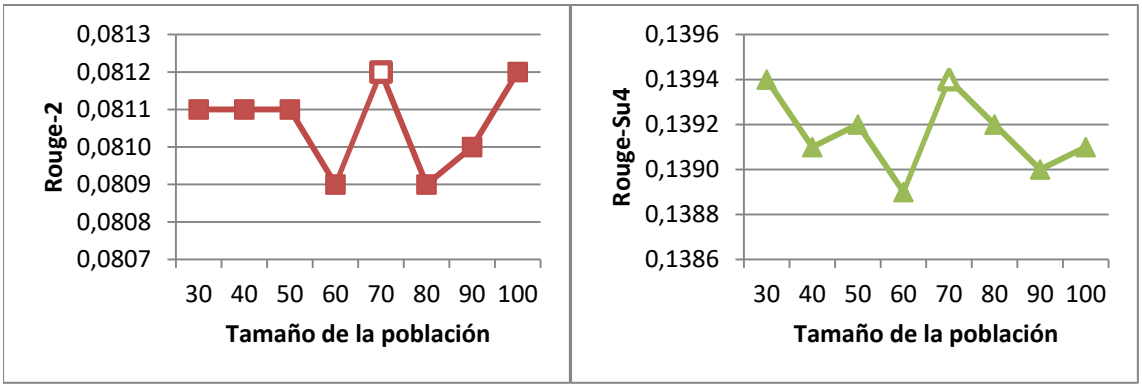

Figura 20. Afinación del tamaño de la población

En la Figura 21 teniendo en cuenta los mejores valores afinados de probabilidad de optimización y tamaño de población se puede observar que se obtuvieron mejores resultados en las medidas de Rouge-2 y Rouge-Su4 con un valor de lambda de 0.86.

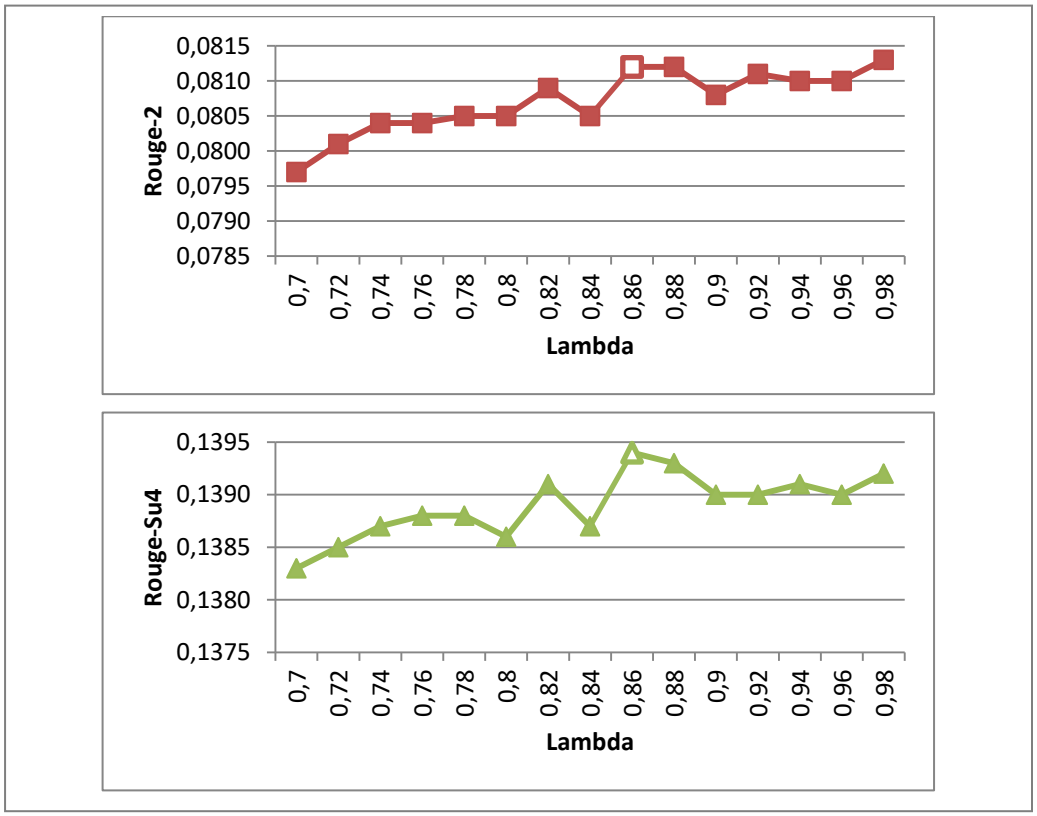

Figura 21. Afinación de lambda

En la Figura 22 teniendo en cuenta los anteriores parámetros afinados se puede observar que se alcanzaron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con una máxima longitud de resumen de 290.

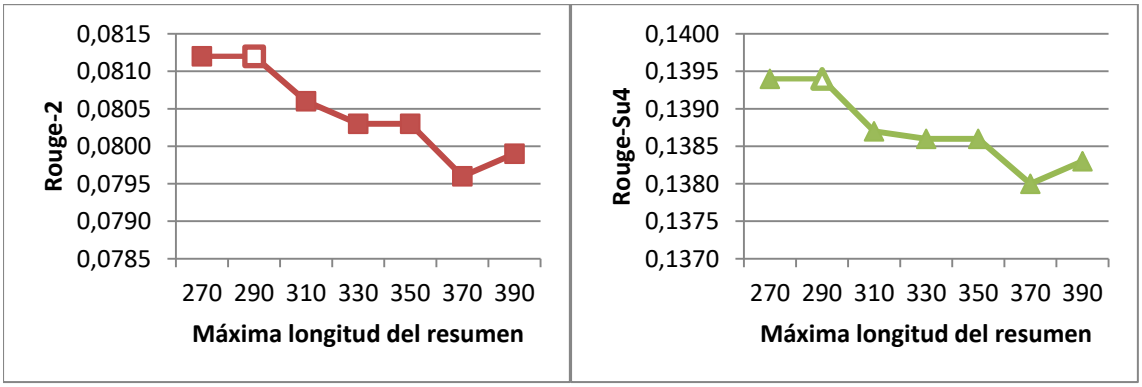

Figura 22. Afinación de máxima longitud del resumen

Al afinar cada uno de los parámetros nombrados de la mejor configuración obtenida con la hiperheurística y hacer su combinación, se puede concluir que se lograron mejores resultados en las medidas de Rouge-2 y Rouge-Su4 con: probabilidad de optimización de

0.5, tamaño de la población 70, lambda 0.86 y máxima longitud del resumen 290 para el conjunto de datos de DUC2005.

4.6.1.2 Afinación para DUC2007

En la Figura 23 se puede observar que se obtuvieron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con un valor de probabilidad de optimización de 0.4.

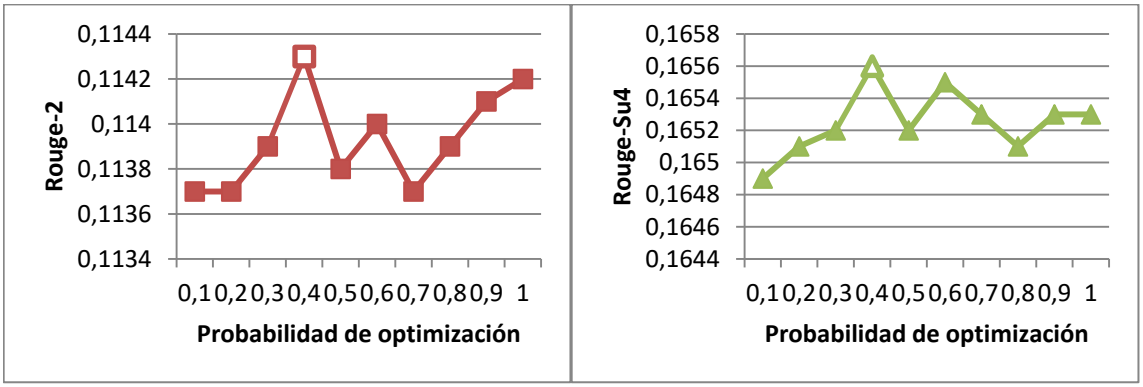

Figura 23. Afinación de la probabilidad de optimización

En la Figura 24 utilizando el mejor valor afinado de la probabilidad de optimización se puede observar que se lograron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con un tamaño de población de 50.

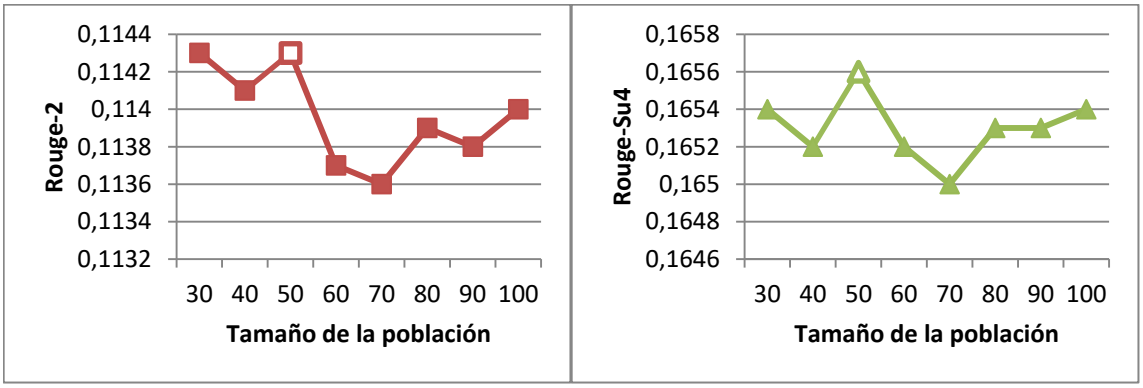

Figura 24.Afinación del tamaño de la población

En la Figura 25 teniendo en cuenta los mejores valores afinados de probabilidad de optimización y tamaño de población se puede observar que se obtuvieron mejores resultados en las medidas de Rouge-2 y Rouge-Su4 con los valores de lambda de 0.80 y 0.86.

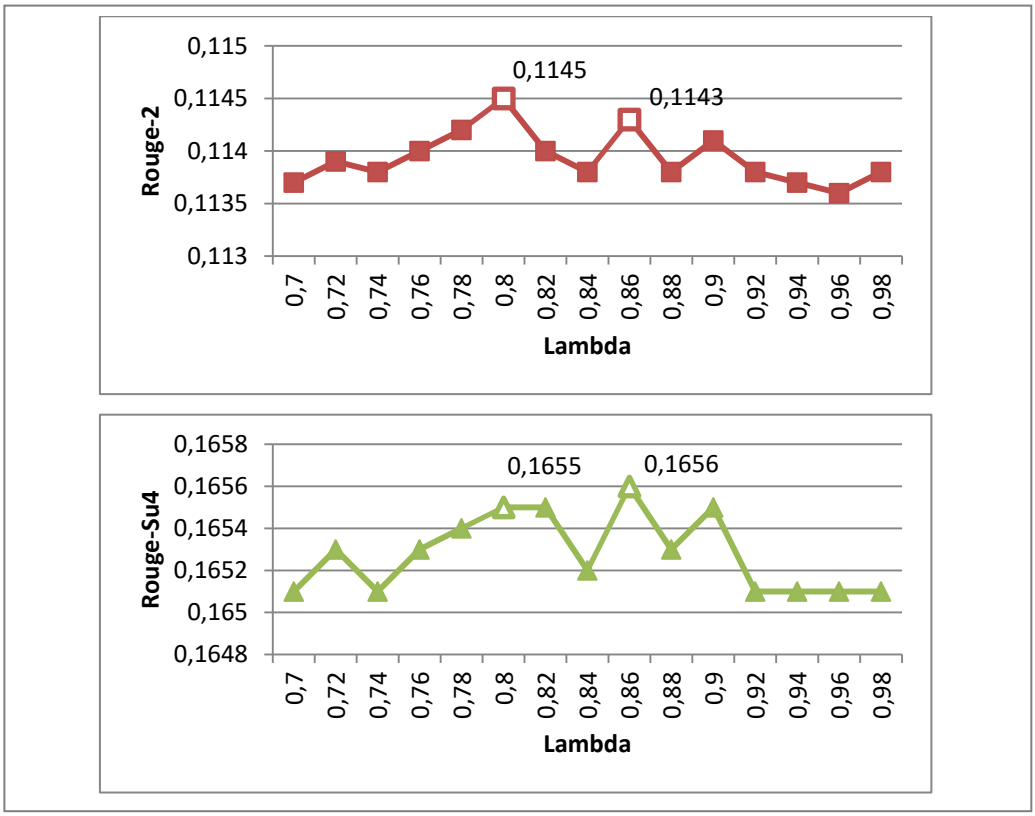

Figura 25. Afinación de Lambda

Como al afinar el valor de lambda se obtuvieron dos valores mejores, entonces se afino la máxima longitud del resumen utilizando primero el valor de lambda de 0.80 y luego el valor de lambda de 0.86, junto con los mejores valores afinados de la probabilidad de optimización y tamaño de la población.

En la Figura 26 se puede observar que se lograron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con una máxima longitud del resumen de 270, teniendo en cuenta el mejor valor de probabilidad de optimización, tamaño de la población y un valor de lambda de 0.80.

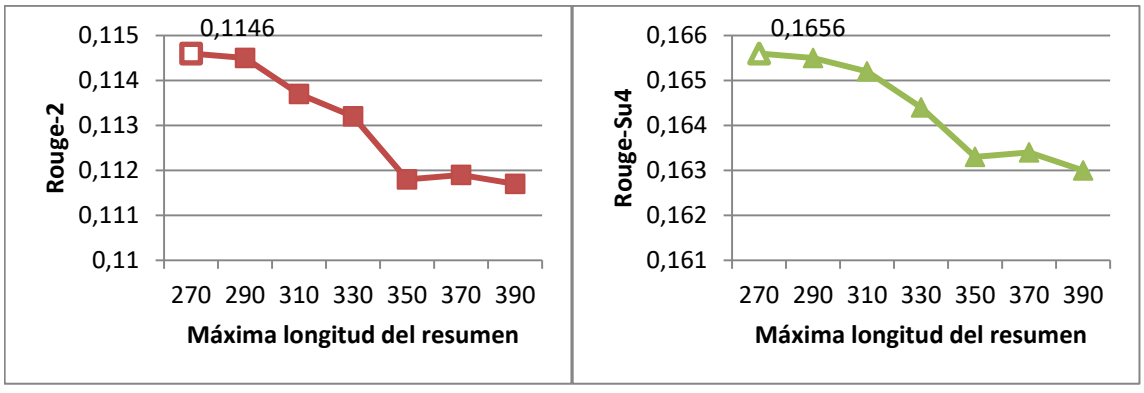

Figura 26. Afinación de máxima longitud del resumen con un valor de lambda de 0.80

En la Figura 27 se puede observar que se obtuvieron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con una máxima longitud del resumen de 270, teniendo en cuenta el mejor valor de probabilidad de optimización, tamaño de la población y un valor de lambda de 0.86.

Figura 27. Afinación de máxima longitud del resumen con un valor de lambda de 0.86

Se puede observar de las Figuras Figura 26 yFigura 27 que los mejores valores en las medidas de Rouge-2 y Rouge-Su4 se obtuvieron con una máxima longitud del resumen de 270, con un valor de lambda de 0.86, tal como se observa en la Figura 27. Además, al afinar cada uno de los parámetros nombrados de la mejor configuración obtenida con la hiperheurística, y hacer su combinación; se puede concluir que se lograron mejores resultados en las medidas de Rouge-2 y Rouge-Su4 con: probabilidad de optimización de 0.4, tamaño de la población 50, lambda 0.86 y máxima longitud del resumen 270 para el conjunto de datos de DUC2007.

4.6.1.3 Mejor afinación para los conjuntos de datos DUC2005 y DUC2007

Se utilizó la mejor afinación de parámetros obtenida con DUC2005 para DUC2007 y viceversa.

	Valores originales		**Nuevos valores**	
Parámetros	DUC2005	DUC2007	DUC2005	DUC2007
Probabilidad de Optimización	0.5	0.4	0.4	0.5
Tamaño de la Población	70	50	50	70
Lambda	0.86	0.86	0.86	0.86
Máxima Longitud del Resumen	290	270	270	290
Medidas				
Rouge-2	0.0812	0.1146	0.8138	0.1140
Rouge-SU4	0.1394	0.1660	0.1394	0.1655

Tabla 3. Comparación de resultados con las valores de las afinaciones de los parámetros originales e intercambio de parámetros afinados

De la Tabla 3 se puede observar que se obtuvieron mejores valores en las medidas de Rouge-2 y Rouge-Su4 con la combinación de parámetros afinados de DUC2007 para ambos conjuntos de datos.

Capítulo 5

5 EVALUACIÓN

5.1 PRE-PROCESAMIENTO DE DOCUMENTOS

Antes de la ejecución del algoritmo propuesto para la generación del resumen, se realizó la etapa de pre-procesamiento, en la cual, se procesa el texto de los documentos para obtener unidades más pequeñas (oraciones), clasificarlo, etiquetarlo y filtrar las palabras u oraciones que podrían constituir ruido en la etapa de selección de las oraciones más representativas del resumen, ésta etapa de pre-procesamiento normalmente genera cierto tipo de dependencia del lenguaje, por ejemplo las palabras vacías deben ser en el idioma específico a resumir. Las etapas de pre-procesamiento usadas en este trabajo son:

5.1.1 Segmentación

El texto original debe ser dividido en unidades pequeñas con el objetivo de realizar la extracción de las oraciones que se van a presentar en el resumen. Es necesario dividir el texto en palabras para poder identificar correctamente los límites entre las unidades de texto u oraciones.

Una de las técnicas para subdividir textos en unidades pequeñas o sub-tópicos se denomina TextTiling [86]. Esta técnica usa patrones de co-ocurrencia léxica y distribución. El algoritmo tiene tres partes principales: pre-procesamiento, cálculo de puntuaciones léxicas e identificación de los límites. En la primera parte se eliminan las palabras vacías, se realiza un análisis morfológico del texto y los documentos se dividen en secuencias de oraciones significativas, sin considerar signos de puntuación. Luego se determina una puntuación léxica para los espacios entre grupos de oraciones, finalmente se realiza una identificación de límites.

Para la segmentación de oraciones en el presente trabajo se hace uso del segmentador diseñado originalmente para AnswerBus Question Answering System y que ahora es utilizado por Seven Tones Search Engine y por muchas otras aplicaciones de PLN. Fue seleccionado debido a que ha sido utilizado previamente por el grupo de investigación GTI (Grupo de Tecnologías de la Información) y se encuentra disponible en http://www.answerbus.com/sentence/.

5.1.2 Filtro de palabras vacías

Las palabras vacías o stopwords son aquellas palabras que son muy frecuentes en un documento pero no contribuyen al contenido del mismo, por ejemplo, palabras como “the”, “about”, “else”, “got”, entre otras; de manera individual no son buenos discriminantes cuando se quiere determinar la relevancia de una unidad de texto en un documento y en la mayoría de los casos constituyen ruido. La eliminación de las palabras vacías puede aumentar la eficiencia del proceso de indización en un 30% a 50% [87]. Una lista de palabras vacías puede ser extendida para incluir artículos, preposiciones y conjunciones y además algunos verbos, adverbios y adjetivos pueden ser tratados como palabras vacías.

La eliminación de palabras vacías reduce considerablemente el tamaño del texto de origen, con el objetivo de calcular los pesos de cada una de las oraciones procesadas (sin palabras vacías). Esta técnica es común y necesaria en el procesamiento del lenguaje natural. Para remover éstas palabras, en este proyecto se tomó una lista de palabras vacías[12] que ha sido usada en el grupo de investigación GTI y en [38].

5.1.3 Stemming

Es una técnica de reducción que permite detectar variantes morfológicas de un mismo término y reemplazarlas por el término raíz o lema. En un texto, la misma palabra usualmente ocurre en muchas variantes morfológicas. Esas formas variantes son gobernadas por el contexto, es decir, si esta se presenta en forma plural o singular, tiempo presente o pasado, etc. En muchos casos, esas diferencias de formas léxicas tienen interpretaciones semánticas diferentes y pueden a menudo ser consideradas como equivalentes para el propósito de procesar mucha información. Para que un sistema de gestión de la información sea capaz de tratar esas formas variantes como un *stem* o *lema*, es común usar un algoritmo de stemming o stemmer, que es un procedimiento computacional que reduce todas las palabras con una misma raíz a una forma común, es decir, si por ejemplo en el texto original se encuentran palabras como *computational* y *computing,* ambas palabras se pueden representar como *comput.*

El efecto no es sólo que diferentes variantes de un término puedan ser llevadas a una forma simple de representación, sino que también reduce el tamaño del vocabulario necesario para la representación del documento o del conjunto de documentos. En muchos casos, la considerable reducción del tamaño del diccionario es útil porque reduce el espacio de almacenamiento, el tiempo de procesamiento, reduce el tamaño de la estructura de indización, así como también hace que la representación del documento sea menos ruidosa y más versátil. Los plurales, los gerundios y los sufijos del tiempo pasado son ejemplos de las variaciones sintácticas que impiden una correspondencia perfecta entre la palabra de una consulta y una palabra respectiva en el documento. Este problema puede ser cubierto con la aplicación del stemming.

Las reglas que forman los algoritmos clásicos de stemming dependen del idioma de las colecciones de los documentos a procesar. El algoritmo clásico es Porter Stemmer[13] cuya versión original está para el idioma inglés [88].

5.1.4 Eliminación de oraciones que tienen similitud menor a un umbral

Cuando se calcula la similitud entre una oración y la colección de documentos, aplicando la ley de cosenos, este valor se encuentra entre 0 y 1, donde cero indica que no tienen términos en común y uno que tiene todos los términos en común. Dado este concepto y lo planteado por Tamayo y Vela [89] donde establecen un umbral para definir si una oración es incluida en el resumen, obteniendo mejores resultados cuando se eliminaban las

[12] ftp://ftp.cs.cornell.edu/pub/smart/english.stop
[13] http://tartarus.org/martin/PorterStemmer/

oraciones que tenían una similitud menor de 0.13; en la presente investigación se decidió usar este valor de umbral.

5.1.5 Lucene

Las etapas anteriores mencionadas para el pre-procesamiento son un proceso fundamental en tareas de procesamiento del lenguaje natural, porque permiten que se haga un procedimiento más preciso del texto. Al realizar las etapas anteriores se va poder calcular el peso de los términos de cada una de las oraciones del texto por medio del esquema de pesado de términos explicado en la sección 2.2.2, peso que va a permitir discriminar entre las oraciones relevantes e irrelevantes en el texto. Además este pesado de términos sirve para calcular la similitud de cosenos. Existen herramientas o bibliotecas de código que realizan algunas de las etapas del pre-procesamiento, muchas de ellas son de Código Abierto, por lo tanto su código fuente está a disposición de la comunidad y puede ser reutilizado en cualquier aplicación. Es importante aclarar que las bibliotecas de funciones no son aplicaciones que se descargan, ejecutan e instalan sino que son APIs (Aplication Programming Interface) a través de las cuales se añaden, con esfuerzos de programación, capacidades de indización y búsqueda a cualquier sistema que se esté desarrollando.

Lucene es una tecnología para la Recuperación de Información que realiza procesos de indización y búsqueda, es creada bajo una metodología orientada a objetos y cuenta con una API implementada en Java, también está disponible en otros lenguajes de programación, soporta la indización de documentos con formatos: txt, pdf, doc, ppt, rtf, xml y html. La principal ventaja de Lucene es su flexibilidad, permite su utilización en cualquier sistema que lleve a cabo procesos de indización o búsqueda, tiene versiones para otros lenguajes como Perl, C#, Ruby y C++. Lucene está disponible bajo la licencia Apache Software Licence en [90]. Además, Lucene posee mejores características en comparación con otras librerías y es la única que utiliza como modelo de representación de los documentos el modelo de espacio vectorial, el cual se considera el más apropiado para el desarrollo del algoritmo memético. Por éstas razones y basados en previos estudios realizados por el grupo de investigación GTI (Grupo de Tecnologías de la Información) acerca de las diferentes herramientas para pre-procesamiento se decidió usar Lucene.

5.2 CORPUS DE EVALUACIÓN

Para la evaluación de los resúmenes generados por el algoritmo memético, se utilizaron los conjuntos de datos de DUC2005 y DUC2007, los cuales pertenecen al dominio periodístico obtenido de TREC[14](The Text Retrieval Conference) y de AQUAINT[15](Advanced Question Answering for Intelligence), que proponen tareas orientadas a resúmenes de múltiples documentos. El contenido de un documento está sujeto a la estructura que se muestra en la Figura 28, la cual es la dispuesta por DUC.

[14] http://trec.nist.gov/overview.html

[15] http://www-nlpir.nist.gov/projects/aquaint/

DUC proporciona los conjuntos agrupados por temas además de varios resúmenes ideales para un mismo conjunto de documentos, esto es, porque no existe un único resumen para un documento o un conjunto de documentos, por lo que proporcionar varios resúmenes ideales permite que el resumen generado automáticamente tenga mayor o menor similitud con alguno de ellos. Todos los documentos fueron segmentados en oraciones, se eliminaron aquellas que tenían similitud con el documento menor a 0.13 y cada una de ellas se le aplica stemming y eliminación de palabras vacías (stopwords) por medio de la librería de Lucene.net.

DUC2005 cuenta con 50 conjuntos de datos y cada conjunto contiene de 25 a 50 documentos, DUC2007 cuenta con 45 conjuntos y cada conjunto con 25 documentos. El algoritmo obtenido genera un resumen de no más de 250 palabras para DUC2005 y DUC2007. En la Tabla 4 se muestra una breve descripción de los datos.

	DUC 2005	**DUC 2007**
Número de grupos	50	45
Número de documentos	1593	1125
Fuente de datos	TREC y TIPSTER	AQUAINT
Longitud del resumen	250 palabras	250 palabras

Tabla 4. Conjunto de datos

```
<DOC>
        <HEADLINE>
        Titulo documento
        </HEADLINE>
        <TEXT>
                <P>
                Texto del documento
                </P>
        </TEXT>
</DOC>
```

Figura 28. Estructura de los documentos

5.3 MÉTRICAS DE EVALUACIÓN

Para la evaluación se usó la herramienta ROUGE 1.5.5 [35] (ver sección 2.1.4.2); para este trabajo se utilizan las medidas ROUGE-2 y ROUGE-SU4, por ser las medidas más usadas para evaluar la calidad de resúmenes de texto generados automáticamente en los artículos del estado del arte con los que se realizó la comparación.

5.4 RESULTADOS Y ANÁLISIS

5.4.1 Resultados del entorno hiperheurístico

El entorno hiperheurístico fue codificado en lenguaje c# .net y se ejecutó en un PC Intel Pentium IV con 1 GB de RAM. Los esquemas de selección de alto y bajo nivel se escogieron teniendo en cuenta la calidad y la diversidad. Lo mismo para los esquemas de cruce y reemplazo. En cuanto a los esquemas de búsqueda local estos son los más utilizados.

Se realizaron dos experimentaciones con diferentes conjuntos de heurísticas de bajo nivel (Ver Anexo E), utilizando los esquemas de selección de alto nivel selección por ruleta y selección por torneo probabilístico explicados en las secciones 3.3.1, con el fin de obtener la mejor configuración de heurísticas de bajo nivel para el algoritmo memético para los conjuntos de datos de DUC2005 y otra para DUC2007.

5.4.1.1 Configuración con el primer conjunto de heurísticas de bajo nivel

Para esta configuración se utilizaron los esquemas de bajo nivel explicados en la sección 3.3.2. En la Tabla 5 Se puede ver las mejores configuraciones que se obtuvieron en cada conjunto de documentos.

Parámetros	**DUC2005**	**DUC2007**
Selección de Alto Nivel	Ruleta	Torneo Probabilístico
Selección	Emparejamiento Restringido	Emparejamiento Restringido
Cruce	Uniforme	Unipunto
Búsqueda Local	Búsqueda de vecindad con Distancia de hamming uno y dos Greedy	Búsqueda de vecindad con Distancia de hamming uno y dos Greedy
Reemplazo	Competencia Restringida	Competencia Restringida

Tabla 5. Mejores configuraciones con el primer conjunto de heurísticas de bajo nivel

5.4.1.2 Configuración con el segundo conjunto de heurísticas de bajo nivel

Para esta configuración se hizo algunas variaciones en los conjuntos de heurísticas de bajo nivel quedando el conjunto de heurísticas de la siguiente forma:

- Selección: ruleta, emparejamiento restringido y basado en rango.
- Cruce: unipunto y uniforme.
- Búsqueda local: Búsqueda de vecindad (con distancia de hamming uno y dos Greedy, y con distancia de hamming uno y dos Aleatorio), en búsqueda local iterada se utilizaron dos algoritmos de búsqueda para la optimización de los agentes (búsqueda por vecindad greedy con distancia de hamming uno y con distancia de hamming dos) y búsqueda local guiada.

En esta configuración se agregó la selección basada en rango y la búsqueda local guiada, debido a los buenos resultados obtenidos en la tesis de pregrado del programa de Ingeniería de Sistemas de la Universidad del Cauca, titulada “Generación Automática de Resúmenes de Un Solo Documento Basada en Algoritmos Meméticos” bajo la misma dirección de la profesora Martha Mendoza. Además se quitaron dos de las búsquedas locales iteradas que se manejaban de forma aleatoria porque éstas dieron bajos resultados en la primera configuración.

En la Tabla 6 se puede ver las mejores configuraciones que se obtuvieron con este segundo conjunto de heurísticas de bajo nivel en cada conjunto de documentos.

Parámetros	**DUC2005**	**DUC2007**
Selección de Alto Nivel	Ruleta	Torneo Probabilístico
Selección	Basado en rango	Emparejamiento Restringido
Cruce	Uniforme	Unipunto
Búsqueda Local	Búsqueda de vecindad con Distancia de hamming uno y dos Greedy	Búsqueda de vecindad con Distancia de hamming uno y dos Greedy
Reemplazo	Competencia Restringida	Peores Individuos

Tabla 6. Mejores configuraciones con el segundo conjunto de heurísticas de bajo nivel

5.4.1.3 Configuración para el algoritmo memético

Luego de realizar las experimentaciones con los dos conjuntos de heurísticas de bajo nivel, se realizaron otros experimentos para escoger la configuración con los mejores resultados en ambos conjuntos (ver Anexo F), debido a que se obtuvieron diferentes mejores configuraciones; seleccionando de esta forma la configuración de DUC2007 obtenida con el segundo conjunto de heurísticas de bajo nivel como se observa en la Tabla 7.

Parámetros	**DUC2007**
Selección	Emparejamiento Restringido
Cruce	Unipunto
Búsqueda Local	Búsqueda de vecindad con Distancia de hamming uno y dos Greedy
Reemplazo	Peores Individuos

Tabla 7. Configuración para DUC2005 y DUC2007

5.4.2 Comparación diferentes métodos

A continuación se da a conocer una pequeña descripción de los métodos con los cuales se compararon los resultados obtenidos con el algoritmo memético propuesto.

- MCMR (B&B) y MCMR (PSO) [38], son métodos no supervisados que para la generación del mejor resumen tienen en cuenta dos propiedades principales: cobertura y redundancia. El primer sistema se basa en el algoritmo de Ramificación y Poda para encontrar la solución óptima. El segundo sistema se basa en el algoritmo de Optimización de Partículas de Enjambre para encontrar la solución óptima.

- TMR +TF (Resumen orientado a Consultas basado en múltiples temas) [91], se basa en un método probabilístico para resolver el problema de generación de resumen de múltiples documentos; este método es orientado a consultas y tiene como objetivo extraer un resumen informativo de una colección de documentos.

- TranSumm (Resumen transductor[16]) [92], este método utiliza un enfoque transductivo que ayuda a identificar dos conjuntos de oraciones: relevantes e irrelevantes a una pregunta.

- Content-term [93], se presenta un método basado en la estimación de los términos del contenido para la generación automática de resúmenes. En el proceso de estimación de los términos del contenido, se usó la función relevante y la función de información de la riqueza de asignación de importancia para cada uno de ellos. Tiene como objetivo producir un resumen con base en una consulta realizada por un usuario.

- PolyU (Universidad Politecnica de Hong Kong en DUC 2005) [94], se presenta un método basado en consultas haciendo uso de la estructura MEAD, tomando sus características para el resumen de texto para múltiples documentos, como: centroide y longitud de la oración; adicionalmente toma otras características: basadas en patrones, términos y semántica. Estas características permiten evaluar la importancia de una frase para incluir en el resumen.

- Biased LexRank [95], es un método semi-supervisado en el contexto de la pregunta y la respuesta para la recuperación de fragmentos. Representa un texto como gráfico de pasos relacionados, con base en su similitud de pares léxica. Utiliza las técnicas tradicionales de recuperación de paso para identificar los pasos que puedan ser relevantes a la consulta en lenguaje natural de un usuario.

- QEA [43], presenta un método de expansión de consulta el cual es combinado con el algoritmo basado en grafos para la generación de resúmenes de múltiples documentos. Para la redundancia aplica una penalización sobre las oraciones.

- Qs-MRC [96], en este método se extiende el refuerzo mutuo (MR, por sus siglas en inglés Mutual Reinforcerment) a la cadena de refuerzo mutuo (MRC, por sus siglas en inglés Mutual Reinforcement Chain) de tres granularidades de texto, es decir documentos, oraciones y términos. El objetivo es proporcionar una estructura de refuerzo general y un modelo matemático para el MRC. Se desarrolla la similitud de consulta sensible para medir la afinidad entre el par de textos.

- SNMF +SLSS [97], es un método basado en el análisis de oraciones nivel semántico (SLSS, por sus siglas en inglés Sentence-Level Semantic Analysis) y factorización de matriz simétrica no negativa (SNMF por sus siglas en inglés Symmetric Non-Negative

[16] Transductor, es un modelo de autómata finito con transiciones sobre parejas de símbolos, que permiten definir relaciones regulares entre lenguajes.

Matrix Factorization). SLSS es capaz de captar las relaciones semánticas entre oraciones y SNMF pueden dividir las oraciones en grupos para la extracción.

- PNRR (por sus siglas en inglés Ranking Sentences with Positive and Negative Reinforcement). Consiste en la clasificación de frases positivas y negativas [98]. Es un algoritmo basado en grafos de clasificación de oraciones para el resumen de actualización.
- GSPSum, por sus siglas en inglés Graph-Based Subtopic Partition Algorithm for Summarization. Es un algoritmo para resumen basado en grafo de partición de subtemas [99].

- LexRank [27], aplican el análisis de grafos y tiene en cuenta la influencia de las otras frases, lo que permite una mejor panorama de las relaciones entre las frases.

- SVR (por sus siglas en inglés Support Vectorial Regression) [100], se utiliza el modelo SVR de forma automática combinando las características y los puntajes de las frases. Se extraen las frases más importantes y son reorganizadas para formar un resumen.

Para la comparación de la mejora del algoritmo propuesto con respecto a otros sistemas se ha usado el cálculo de la mejora relativa $\frac{(método\ propuesto - otro\ método)}{otro\ método}\ x\ 100$.

5.4.3 Evaluación con DUC 2005

En la Tabla 8 se presentan los resultados de las pruebas de evaluación de los 50 conjuntos de documentos de DUC2005 realizadas con el Algoritmo Memético (AM) con 15000 evaluaciones de la función objetivo, con el objeto de medir la calidad de los resúmenes con respecto a MCMR (PSO) dado que en éste también hace uso de un algoritmo evolutivo cuyo número de evaluaciones de la función objetivo es similar. Las pruebas indican que el algoritmo memético supera a MCMR (PSO) en las medidas de Rouge-2 en un 7.68% y en Rouge-Su4 en un 2.5%. Al igual que el método QEA en un 8.41% y 4.58% en las medidas de Rouge-2 y Rouge-Su4, respectivamente. La comparación también con TMR+TF muestra que el AM mejora el rendimiento en las medidas de Rouge-2 en un 13.57% y en Rouge-Su4 en un 6,90%. Esto indica que la configuración obtenida con el enfoque hiperheurístico tiene un buen desempeño en generación de resúmenes de múltiples documentos. En esta tabla también se muestran los resultados obtenidos con otros métodos del estado del arte.

Algoritmo	ROUGE-2	ROUGE-SU4
AM(15.000)	**0,0812**	**0,1394**
AM(7.500)	0,0807	0,1389
MCMR(B&B)	0,0790	0,1392
MCMR(PSO)[38]	0,0754	0,1360
TMR + TF [91]	0,0715	0,1304
TranSumm [92]	0,0755	0,1366
Content-term [93]	0,0718	0,1338
PolyU [94]	0,0717	0,1297
Biased LexRank [95]	0,0753	0,1363
QEA [43]	0,0749	0,1333
Qs-MRC [96]	0,0779	0,1366

SNMF +SLSS [97]	0,0604	0,1230

Tabla 8. Resultados de ROUGE para DUC2005

En la Tabla 9, se puede ver que existe una mejora de AM con 15000 evaluaciones de la función objetivo con respecto a AM con 7500 evaluaciones de la función objetivo. Por lo tanto el desempeño del algoritmo AM depende de la cantidad de evaluaciones de la función objetivo que se ejecuten. También se puede observar que para los valores obtenidos en ROUGE-2 y ROUGE-SU4, AM (15000) supera a todos los demás métodos. También se puede notar que aunque se requieren 15000 evaluaciones de la función objetivo para superar los resultados de MCMR (B&B) en ROUGE-SU4, basta con realizar 7500 evaluaciones de la función objetivo para superar a los demás métodos.

Métodos	Mejora del método AM (%)	
	ROUGE-2	ROUGE-SU4
AM (15.000)	0	0
AM (7.500)	0,61	0,35
MCMR (B&B)	2,78	0,14
MCMR (PSO)	7,69	2,5
TMR + TF	13,57	6,90
TranSumm	7,54	2,05
Content-term	13,09	4,18
PolyU	13,25	7,48
Biased LexRank	7,83	2,27
QEA	8,41	4,58
Qs-MRC	4,23	2,05
SNMF +SLSS	34,43	13,34

Tabla 9. Comparación AM con otros métodos para conjuntos de documentos DUC2005.

5.4.4 Evaluación con DUC 2007

En la Tabla 10 se presentan los resultados de las pruebas de evaluación de los 45 conjuntos de documentos de DUC 2007 realizadas con el Algoritmo Memético (AM) con 15000 evaluaciones de la función objetivo, con el objeto de medir la calidad de los resúmenes con respecto a MCMR (PSO). Las pruebas indican que el algoritmo memético está por debajo de MCMR (PSO) en las medidas de Rouge-2 y en Rouge Su4. A pesar de esto, los resultados obtenidos para AM (15000) son promisorios ya que obtiene mejores resultados que los otros métodos del estado del arte. A pesar de esto se puede observar que las pruebas con el algoritmo memético superan al método PNR^2 en las medidas de Rouge-2 y Rouge-Su4 en un 28.04% y 28.58% respectivamente. Al igual que el método LexRank en un 16.10% y 11.93% en las medidas de Rouge-2 y Rouge-Su4, respectivamente.

Algoritmo	ROUGE-2	ROUGE-SU4
AM(15.000)	0,1146	0,1660

AM(7.500)	0,1144	0,1656
MCMR(PSO) [38]	**0,1165**	**0,1697**
PNR^2 [98]	0,0895	0,1291
GSPSum [99]	0,1110	0,1638
LexRank [27]	0,0987	0,1487
SVR [100]	0,1117	0,1628

Tabla 10. Resultados de ROUGE para DUC2007

Se puede apreciar en la Tabla 11, para los valores obtenidos en ROUGE-2 y ROUGE-SU4, AM con 15000 evaluaciones de la función objetivo supera a seis métodos, pues el valor negativo que se obtiene al realizar la comparación con MCMR (PSO) indica que éstos superan los resultados de AM (15000). También se puede notar que basta con realizar 7500 evaluaciones de la función objetivo para superar a los métodos de PNR^2, GSPSum, LexRank y SVR en las medidas de ROUGE-2 y ROUGE-SU4.

Métodos	Mejora del método AM (%)	
	ROUGE-2	ROUGE-SU4
AM (15.000)	0	0
AM (7.500)	0,17	0,24
MCMR (PSO)	-1,63	-2,18
PNR^2	28,04	28,58
GSPSum	3,24	1,34
LexRank	16,10	11,63
SVR	2,59	1,96

Tabla 11. Comparación AM con otros métodos para conjuntos de DUC2007

Los resultados de la experimentación sobre el conjunto de datos de DUC2005 superaron MCMR (PSO) y otros sistemas del estado del arte; con el conjunto de datos de DUC2007 también se superaron a otras referencias del estado del arte, pero no al sistema MCMR basado en un algoritmo PSO. Los resultados obtenidos de la experimentación con los conjuntos de datos de DUC2005 y DUC2007 muestran que el algoritmo obtenido con el enfoque hiperheurístico para la generación automática de resúmenes obtiene mejores resultados con un conjunto. Esto mismo ocurre para sistemas que utilizan: conjuntos de datos de DUC2002 y DUC2004 para: múltiples documentos basados en un algoritmo de evolución diferencial [101], basado en un algoritmo adaptativo de evolución diferencial eliminando la redundancia mientras selecciona las frases representativas [102]; conjuntos de datos DUC2002, DUC2004 y DUC2006 basado en un algoritmo de evolución diferencial modelando la generación de resúmenes de múltiples documentos como un problema de programación booleana cuadrática [103]; y conjuntos de datos DUC2005 y DUC2006 para: múltiples documentos basados en grafos para la clasificación de oraciones [43], resumen orientado a consultas basado en múltiples temas [91]. Otra razón que puede influir en estos resultados es la función objetivo, esto se observó en una evaluación adicional con el algoritmo memético propuesto con la función objetivo de MCMR (PSO), cuyos resultados con las medidas de ROUGE-2 y ROUGE-SU4 disminuyeron en comparación a los resultados con nuestra función objetivo, pero también se observó que los resultados fueron mejores para el conjunto de datos de DUC2005 que con el conjunto de DUC2007; esto implica que los factores o variables que se incluyen en

la función objetivo hacen que los resultados sean mejores o peores, por el cual esto también es un área de investigación actualmente.

5.4.5 Comportamiento del AM con diferentes evaluaciones de la función objetivo.

5.4.5.1 Evaluación con DUC2005

En la Tabla 12 se presentan los resultados de las pruebas de evaluación de los 50 conjuntos de documentos de DUC2005 realizadas con el AM con 7500, 20000 y 25000 evaluaciones de la función objetivo, con el objeto de observar el comportamiento de estas medidas de ROUGE al incrementar la cantidad de evaluaciones de la función objetivo.

Evaluaciones de la función objetivo Algoritmo memético (AM)	ROUGE-2	ROUGE-SU4
7500	0,0808	0,1389
15000	0,0812	0,1394
20000	0,0814	0,1396
25000	**0,0815**	**0,1396**

Tabla 12. Resultados de Rouge con diferentes evaluaciones de la función objetivo del AM para DUC2005

Las pruebas indican que a medida que se incrementa la cantidad de evaluaciones de la función objetivo, las medidas de ROUGE van mejorando, obteniendo mejores resultados con AM (25000). También se puede observar que a partir de 20000 evaluaciones de la función objetivo la medida de ROUGE-SU4 se mantiene constante.

5.4.5.2 Evaluación con DUC2007

En la Tabla 13 se presentan los resultados de las pruebas de evaluación de los 45 conjuntos de documentos de DUC2007 realizadas con el AM con 7500, 20000 y 25000 evaluaciones de la función objetivo, con el objeto de conocer si se comporta similar a los resultados de la Tabla 12.

Evaluaciones de la función objetivo Algoritmo memético (AM)	ROUGE-2	ROUGE-SU4
7500	0,1144	0,1657
15000	0,1146	0,1660
20000	**0,1149**	**0,1660**
25000	0,1148	0,1659

Tabla 13. Resultados de Rouge con diferentes evaluaciones de la función objetivo del AM para DUC2007

Se puede apreciar en la Tabla 13 que se obtiene una mejora de los resultados respecto a la medida de ROUGE-2 con el AM (20000) superando al AM (7500) y AM (15000). Al parecer después de 20000 evaluaciones de la función objetivo las medidas de ROUGE tienden a disminuir.

Capítulo 6

6 CONCLUSIONES Y TRABAJO FUTURO

6.1 CONCLUSIONES

En este trabajo se propone un enfoque hiperheurístico para encontrar un algoritmo memético para generación de resúmenes de múltiples documentos; que permitió encontrar la mejor configuración entre los esquemas de selección por ruleta y por emparejamiento restringido; cruce unipunto y uniforme; búsqueda local iterada y por vecindad variable; reemplazo por competencia restringida y de los peores individuos. La función objetivo a optimizar está compuesta por un factor de cobertura, que mide la similitud entre el texto de las oraciones del resumen candidato y el centroide de las oraciones de la colección de documentos; y un factor de redundancia que mide que tan similares son las oraciones que componen el resumen candidato.

El algoritmo memético que se obtuvo desde el enfoque hiperheurístico propuesto, está definido por el esquema de selección de emparejamiento restringido, cruce unipunto, búsqueda por vecindad greedy con distancia de hamming 1 y 2 para optimizar los descendientes y para actualizar la población el reemplazo por peores individuos.

La configuración encontrada para el algoritmo memético por medio del enfoque hiperheurístico, presentó variaciones en algunos de los esquemas para los conjuntos de datos de DUC2005 y DUC2007, para obtener una única configuración del algoritmo memético se escogió la que permitió obtener los mejores resultados en ambos conjuntos de datos, esta fue la obtenida para el conjunto de datos de DUC2007.

El proceso de afinación de los parámetros de probabilidad de optimización, tamaño de la población, lambda y máxima longitud del resumen permitió obtener mejores resultados en las medidas de ROUGE-2 y ROUGE-SU4 sobre los conjuntos de DUC2005 y DUC2007.

El algoritmo memético obtenido para el conjunto de datos de DUC2005 supera a MCMR (PSO) en las medidas de Rouge-2 en un 7.68% y en Rouge-Su4 en un 2.5%. Al igual que el método QEA en un 8.41% y 4.58% en las medidas de Rouge-2 y Rouge-Su4, respectivamente. Además, la comparación con TMR+TF muestra que el AM mejora el rendimiento en las medidas de Rouge-2 en un 13.57% y en Rouge-Su4 en un 6,90%.

El algoritmo memético obtenido para el conjunto de datos de DUC2007 supera al método PNR en las medidas de Rouge-2 y Rouge-Su4 en un 28.04% y 28.58% respectivamente. Al igual que al método LexRank en un 16.10% y 11.93% en las medidas de Rouge-2 y Rouge-Su4, respectivamente.

El algoritmo memético obtenido desde el enfoque hiperheurístico se evaluó por medio de las medidas ROUGE-2 y ROUGE-SU4 sobre los conjuntos de datos de DUC2005 y DUC2007; al realizar la comparación del algoritmo propuesto con MCMR (PSO) se superaron los resultados para DUC2005. También se superaron los resultados con

respecto a los dos conjuntos de datos con diferentes métodos basados en grafos, reducción algebraica y aprendizaje de máquina. Los resultados obtenidos por el algoritmo memético propuesto en este proyecto con los conjuntos de datos de DUC2005 y DUC2007 son promisorios ya que superan diferentes métodos del estado del arte.

6.2 RECOMENDACIONES Y TRABAJO FUTURO

Como trabajo futuro se espera incluir dentro del entorno hiperheurístico otros esquemas de selección, cruce y búsqueda local, para tratar de obtener mejores resultados que los presentados en este proyecto.

Evaluar el algoritmo con otros conjuntos de datos de las conferencias de DUC y TAC, para analizar el comportamiento del algoritmo memético en la tarea de generación automática de múltiples documentos.

Afinar los parámetros utilizados con otra técnica, que permita mejorar la exploración del espacio de búsqueda de estos parámetros y de esta forma encontrar la mejor combinación de valores con respecto a la probabilidad de optimización, tamaño de la población, lambda y máxima longitud del resumen; de esta forma conseguir mejores resultados en las medidas de Rouge. Además realizar la afinación de estos parámetros teniendo en cuenta la unión de los dos conjuntos de documentos DUC, para tratar de generalizar el algoritmo.

Explorar otros factores en la función objetivo u otras fórmulas matemáticas para los factores de cobertura y redundancia de esta propuesta, que permitan obtener mejores resultados de los resúmenes generados automáticamente.

Capítulo 7

7 BIBLIOGRAFÍA

[1] C. Munehs, G. Vikrant, and P. Santosh Kr, "A Statistical Approach for Automatic Text Summarization by Extraction," in *Proceedings of the 2011 International Conference on Communication Systems and Network Technologies*: IEEE Computer Society, 2011, pp. 268 - 271.

[2] P. B. Baxendale, "Machine-made index for technical literature: an experiment," *IBM J. Res. Dev.,* vol. 2, pp. 354-361, 1958.

[3] M. C. John and P. O. l. Dianne, "Text summarization via hidden Markov models," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* New Orleans, Louisiana, United States: ACM, 2001.

[4] L. Chin-Yew and H. Eduard, "Identifying topics by position," in *Proceedings of the fifth conference on Applied natural language processing* Washington, DC: Association for Computational Linguistics, 1997.

[5] O. Miles, "Using maximum entropy for sentence extraction," in *Proceedings of the ACL-02 Workshop on Automatic Summarization - Volume 4* Phildadelphia, Pennsylvania: Association for Computational Linguistics, 2002.

[6] E. R. Barzilay, M, "Using Lexical Chains for Text Summarization," *In Proceedings of the ACL/EACL'97 Workshop on Intelligent Scalable Text Summarization, Madrid, Spain,* pp. 10-17, 1997.

[7] T. R. Mihalcea, P, "TextRank: Bringing Order into Texts," *in Proceeding of the Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain,* 2004.

[8] T. K. Landauer, Dumais, S.T, "A solution to platos problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge," *In Psychological Review,* pp. 211–240, 1997.

[9] V. Qazvinian, L. S. Hassanabadi, and R. Halavati, "Summarising text with a genetic algorithm-based sentence extraction," *International Journal of Knowledge Management Studies,* vol. 2, pp. 426-444, 2008.

[10] P. k. dehkordi, D. F. kumarci, and D. H. khosravi, "Text Summarization Based on Genetic Programming " *International Journal of computing and ICT Research,* vol. 3, pp. 57-64, 2009.

[11] E. Shareghi and L. S. Hassanabadi, "Text Summarization with Harmony Search Algorithm-Based Sentence Extraction " *5th international conference on Soft computing as transdisciplinary science and technology,* 2008.

[12] N. S. Mohammed Salem Binwahlan, Ladda Suanmali, "Swarm Based Text Summarization," 2009.

[13] X.-l. W. by Yan-min Chen, Bing-quan Liu, "Multi-document summarization based on lexical chains," *in Proceedings of 2005 International Conference on Machine Learning and Cybernetics.,* vol. 3, pp. 1937-1942 2005.

[14] T. R. Mihalcea, P, "An Algorithm for Language Independent Single and Multiple Document Summarization," *In Proceedings of the International Joint Konference on Natural Language Processing, Korea,* 2005.

[15] G. M. B. Hachey, and D. Reitter, "The Embra System at DUC 2005: Query-oriented Multi-document Summarization with a Very Large Latent Semantic Space," *n Proceedings of the Document Understanding Conference (DUC), Vancouver, Canada,* 2005.

[16] J. Steinberger and M. Křišťan, "LSA-Based Multi-Document Summarization," *in Proceedings of 8th International PhD Workshop on Systems and Control, Balatonfured, Hungary,* 2007.

[17] R. R. Dragomir, J. Hongyan, Ma, S. gorzata, and T. Daniel, "Centroid-based summarization of multiple documents," *Inf. Process. Manage.,* vol. 40, pp. 919-938, 2004.

[18] L. Hennig, "Topic-based Multi-Document Summarization with Probabilistic Latent Semantic Analysis," *in International Conference RANLP, Borovets, Bulgaria,* 2009.

[19] R. M. A. Rasim M. Alguliev, Makrufa S. Hajirahimova, Chingiz A. Mehdiyev, "MCMR: Maximum coverage and minimum redundant text summarization model," 2011.

[20] Salwani Abdullah, Nasser R. Sabar, Mohd Zakree Ahmad Nazri, and B. M. Hamza Turabieh, "A Constructive Hyper-heuristics for Rough Set Attribute Reduction," *n proceeding of: 10th International Conference on Intelligent Systems Design and Applications, ISDA 2010, November 29 - December 1, 2010, Cairo, Egypt,* pp. 1032-1035, 2010.

[21] E. K. Burke, H. Matthew, K. Graham, O. Gabriela, O. Ender, and Q. Rong, "A Survey of Hyper-heuristics," pp. 1-44, 2009.

[22] C. Cotta Porras and P. Moscato, "Una Introducción a los Algoritmos Meméticos," *Revista Iberoamericana de Inteligencia Artificial,* vol. 7, Nº. 19, 2003, pp. 131-148, 2003.

[23] K. S. Jones, "Automatic Summarising: Factors and Directions," *Advances in Automatic Text Summarization,* 1999.

[24] K. Jezek and S. Josef, "Automatic Text Summarization (The State of the art 2007 and new challenges)," pp. 1-12, 2007.

[25] M. Mieskes, V. Nastase, S. P. Ponzetto, and M. Strube, "Cascaded Filtering for Topic-Driven Multi-Document Summarization," *Proceedings of the Document Understanding Conference 2005,* 2007.

[26] G. Erkan and D. R. Radev, "LexPageRank: Prestige in Multi-Document Text Summarization," 2004.

[27] E. Gunes and R. R. Dragomir, "LexRank: graph-based lexical centrality as salience in text summarization," *J. Artif. Int. Res.,* vol. 22, pp. 457-479, 2004.

[28] D. R. Radev, H. Jing, and M. Budzikowska, "Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies," *NAACL-ANLP 2000 Workshop on Automatic summarization,* pp. 21-30, 2000.

[29] B. Thomas and ck, *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*: Oxford University Press, 1996.

[30] I. Hatzilygeroudis, J. Prentzas, A. Bossard, and C. Rodrigues, "Combining a Multi-Document Update Summarization System –CBSEAS– with a Genetic Algorithm," in *Combinations of Intelligent Methods and Applications*. vol. 8, R. J. Howlett and L. C. Jain, Eds.: Springer Berlin Heidelberg, 2011, pp. 71-87.

[31] C. Aone, M. E. Okurowski, J. Gorlinsky, and B. Larsen, "A Trainable Summarizer with Knowledge Acquired from Robust NLP Techniques," *Advances in Automatic Text Summarization,* pp. 71-80, 1999.

[32] K. S. Jones and J. R. Galliers, "Evaluating natural language processing systems: An analysis and review." vol. 24: Springer Verlag, 1995.

[33] M. Hassel, "Resource Lean and Portable Automatic Text Summarization," in *Computer Science and Communication*. vol. Doctoral Stockholm, Sweden, 2007, p. 144.

[34] C. Y. Lin, "Rouge: A package for automatic evaluation of summaries," 2004, pp. 25-26.

[35] C. Lin, "Rouge: a package for automatic evaluation of summaries," in *Proceedings of the Workshop on Text Summarization Branches Out*, Barcelona, Spain, 2004.

[36] A. P. Porrata, R. B. Llavori, and J. R. Shulcloper, "Desarrollo de Algoritmos para la Estructuración Dinámica de Información y su Aplicación a la Detección de Sucesos," *Castellón, España,* 2004.

[37] S. Gerard, *Automatic text processing: the transformation, analysis, and retrieval of information by computer*: Addison-Wesley Longman Publishing Co., Inc., 1989.

[38] R. M. Alguliev, R. M. Aliguliyev, M. S. Hajirahimova, and C. A. Mehdiyev, "MCMR: Maximum coverage and minimum redundant text summarization model," *Expert Systems with Applications,* vol. In Press, Corrected Proof, 2011.

[39] W. Song, L. Cheon Choi, S. Cheol Park, and X. Feng Ding, "Fuzzy evolutionary optimization modeling and its applications to unsupervised categorization and extractive summarization," *Expert Systems with Applications,* vol. 38, pp. 9112-9121.

[40] M. M. Ali, M. K. Ghosh, and A. Al-Mamun, "Multi-document Text Summarization: SimWithFirst Based Features and Sentence Co-selection Based Evaluation," in *Future Computer and Communication, 2009. ICFCC 2009. International Conference on*, 2009, pp. 93-96.

[41] W. Xiaojun, "An exploration of document impact on graph-based multi-document summarization," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing* Honolulu, Hawaii: Association for Computational Linguistics, 2008.

[42] B. Danushka, O. Naoaki, and I. Mitsuru, "A machine learning approach to sentence ordering for multidocument summarization and its evaluation," in

Proceedings of the Second international joint conference on Natural Language Processing Jeju Island, Korea: Springer-Verlag, 2005.

[43] L. Zhao, L. Wu, and X. Huang, "Using query expansion in graph-based approach for query-focused multi-document summarization," *Information Processing & Management,* vol. 45, pp. 35-41, 2009.

[44] S. H. Zanakis, J. R. Evans, and A. A. Vazacopoulos, "Heuristic methods and applications: A categorized survey," *European Journal of Operational Research,* vol. 43, pp. 88-110, 1989.

[45] E. K. Burke, M. R. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and J. R. Woodward, "Exploring Hyper-heuristic Methodologies with Genetic Programming," in *Computational Intelligence*. vol. 1, C. L. Mumford and L. C. Jain, Eds.: Springer, 2009, pp. 177-201.

[46] A. García-Villoria, S. Salhi, A. Corominas, and R. Pastor, "Hyper-heuristic approaches for the response time variability problem," *European Journal of Operational Research,* vol. 211, pp. 160-169, 2011.

[47] P. Rattadilok, "An Investigation and Extension of a Hyper-heuristic Framework," *Slovenian Society Informatika,* pp. 523–534, 2009.

[48] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward, "A Classification of Hyper-heuristic Approaches," *International Series in Operations Research & Management Science,* vol. 146, pp. 449-468, 2010.

[49] K. Chakhlevitc and P. Cowling, "Hyperheuristics: Recent developments," *Studies in Computational Intelligence* vol. 136, pp. 3-29, 2008.

[50] I. C. Peter, K. Graham, and S. Eric, "A Hyperheuristic Approach to Scheduling a Sales Summit," in *Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling III*: Springer-Verlag, 2001.

[51] I. C. Peter, K. Graham, and S. Eric, "A parameter-free hyperheuristic for scheduling a sales summit," *In Proceedings of the 3rd Metaheuristic International Conference (MIC 2001),* pp. 127–131, 2001.

[52] P. Cowling and K. Chakhlevitch, "Hyperheuristics for managing a large collection of low level heuristics to schedule personnel," in *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*, 2003, pp. 1214-1221 Vol.2.

[53] R. H. Storer, S. D. Wu, and R. Vaccari, "Problem and heuristic search space strategies for job shop scheduling," *ORSA Journal on Computing,* 1995.

[54] D. Beasley, D. R. Bull, and R. R. Martin, "An overview of genetic algorithms: Part 1, fundamentals," *University Computing,* pp. 58-69, 1993.

[55] F. Rothlauf, "Representations for Genetic and Evolutionary Algorithms," Springer-Verlag, Ed. Berlín Heidelberg, 2006.

[56] B. L. M. Brad L. Miller , David E. Goldberg , David E. Goldberg, "Genetic Algorithms, Tournament Selection, and the Effects of Noise," *Complex Systems,* vol. 9, pp. 193-212, 1995.

[57] V. D. Yannibelli, "Algoritmos Genéticos y Meméticos," 2007.

[58] S.N.Sivanandam and S.N.Deepa, "Introduction to Genetic Algorithms," S. B. Heidelberg, Ed. New York, 2008.

[59] R. Sivaraj and T. Ravichandran, "A Review of Selection Methods in Genetic Algorithm," *International Journal of Enginneering Science and Techno logy (IJEST),* vol. 3, 2011.

[60] M. Melanie, *An Introduction to Genetic Algorithms*. London, England, 1998.

[61] A. D. J. Kenneth and M. S. William, "An Analysis of the Interacting Roles of Population Size and Crossover in Genetic Algorithms," in *Proceedings of the 1st Workshop on Parallel Problem Solving from Nature*: Springer-Verlag, 1991.

[62] R. Poli and W. B. Landong, "Schema Theory for Genetic Programming with One-point Crossover and Point Mutation " University of Birmingham, UK 1998.

[63] W. M. Spears and K. A. D. Jong, "On the Virtues of Parameterized Uniform Crossover," *In Proceedings of the Fourth International Conference on Genetic Algorithms,* pp. 230-236, 1991.

[64] H. Xiao-Bing and P. Ezequiel Di, "An efficient genetic algorithm with uniform crossover for air traffic control," *Comput. Oper. Res.,* vol. 36, pp. 245-259, 2009.

[65] M. G. Pose, "Introducción a los algoritmos genéticos," Tecnologías de la Información y Comunicaciones. Universidad de Coruña, Coruña 2000.

[66] O. M. Shir, "Niching in Derandomized Evolution Strategies and its Applications in Quantum Control," Leiden, 2008, p. 256.

[67] P. Hansen, N. Mladenovié, and J. A. M. Pérez, "Búsqueda de Entorno Variable," 2003.

[68] O. C. M. H. R. Lourenco, and T. Stützle, "Iterated Local Search," *Handbook of Metaheuristics,* vol. 7, pp. 321-353, 2003.

[69] C. Voudouris and E. Tsang, "Guided Local Search," University of Essex, Colchester, Technical Report CSM-247 1995.

[70] F. Glover, "Tabu Search Fundamentals and uses," University of Colorado, Boulder, Colorado 1995.

[71] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science,* vol. 220, 1983.

[72] T. A. Feo and M. G. C. Resende, "Greedy Randomized Adaptive Search Procedures," *Journal of Global Optimization,* vol. 6, pp. 109-133, 1995.

[73] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: model, taxonomy, and design issues," *Evolutionary Computation, IEEE Transactions on,* vol. 9, pp. 474-488, 2005.

[74] P. Merz and B. Freisleben, "A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, 1999, p. 2070 Vol. 3.

[75] G. C. Onwubolu and B. V. Babu, "New Optimization Techniques in Engineering," vol. 141, 2004.

[76] F. Neri, C. Cotta, P. Moscato, and J.-K. Hao, "Memetic Algorithms in Discrete Optimization," in *Handbook of Memetic Algorithms*. vol. 379: Springer Berlin Heidelberg, 2011, pp. 73-94.
[77] A. H. Wright, "Genetic algorithms for real parameter optimization," *Foundations of Genetic Algorithms,* 1991.
[78] D. Beasley, D. R. Bull, and R. R. Martin, "An Overview of Genetic Algorithms : Part 1, Fundamentals," *University Computing,* vol. 15, pp. 58-69, 1993.
[79] P. Hansen and N. MladenoviÄ‡, "Variable neighborhood search: Principles and applications," *European Journal of Operational Research,* vol. 130, pp. 449-467, 2001.
[80] K. Erwin, "Matemáticas Avanzadas para Ingeniería." vol. 1 México: Limusa Wiley, 2003., p. 721.
[81] R. Kumar and Jyotishree, "Blending Roulette Wheel Selection & Rank Selection in Genetic Algorithms " *International Journal of Machine Learning and Computing,* vol. 2, pp. 365-370, 2012.
[82] C. Cobos, C. Montealegre, M. F. Mejía, M. Mendoza, and E. León, "Web Document Clustering based on a New Niching Memetic Algorithm, Term-Document Matrix and Bayesian Information Criterion " *Evolutionary Computation (CEC), 2010 IEEE Congress on,* pp. 1-8, 2010.
[83] M. Ryan, "A study of global inference algorithms in multi-document summarization," in *Proceedings of the 29th European conference on IR research* Rome, Italy: Springer-Verlag, 2007.
[84] G. R. Saggion H, "Multi-document summarization by cluster/profile relevance and redundancy removal," in *Proceedings of the Document Understanding Conference 2004* Boston, USA: NIST, 2004.
[85] B. Hachey, G. Murray, and D. Reitter, "The embra system at duc 2005: Query-oriented multi-document summarization with a very large latent semantic space (2005)," *in Proceedings of the Document Understanding Conference (DUC) 2005,* 2005.
[86] M. A. Hearst, "TextTiling: Segmenting text into multi-paragraph subtopic passages," *Computational Linguistics,* vol. 23, pp. 33-64, 1997.
[87] I. A. El-Khair, "Effects of Stop Words Elimination for Arabic Information Retrieval: A Comparative Study," *International Journal of Computing & Information Sciences,* vol. 4, pp. 119-133, 2006.
[88] M. F. Porter, "An algorithm for suffix stripping," Program, 1980.
[89] T. M. W. Andres and V. C. M. Lynnette, "Generación automática de Resúmenes de Múltiples Documentos Basada en el Algoritmo GHS+LEM," in *Departamento de Sistemas* Popayán, Colombia: Universidad del Cauca, 2010.
[90] Lucene, "Sitio web de Lucene: Disponible en http://lucene.apache.org."
[91] L. Y. Jie Tang, Dewei Chen, "Multi-topic based Query-oriented Summarization," *SIAM International Conference Data Mining,* 2009.
[92] R. A. Massih and U. Nicolas, "Incorporating prior knowledge into a transductive ranking algorithm for multi-document summarization," in

Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval Boston, MA, USA: ACM, 2009.

[93] H. Tingting, S. Wei, L. Fang, Y. Zongkai, and M. Liang, "The Automated Estimation of Content-Terms for Query-Focused Multi-document Summarization," in *Proceedings of the 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery - Volume 05*: IEEE Computer Society, 2008.

[94] W. Li, W. Li, B. Li, Q. Chen, and M. Wu, "The Hong Kong Polytechnic University at DUC2005," *Proceedings of the Document Understanding Conference 2005,* 2005.

[95] O. Jahna, E. Gunes, and R. R. Dragomir, "Biased LexRank: Passage retrieval using random walks with question-based priors," *Inf. Process. Manage.,* vol. 45, pp. 42-54, 2009.

[96] W. Furu, L. Wenjie, L. Qin, and H. Yanxiang, "Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* Singapore, Singapore: ACM, 2008.

[97] W. Dingding, L. Tao, Z. Shenghuo, and D. Chris, "Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* Singapore, Singapore: ACM, 2008.

[98] W. Li, F. Wei, Q. Lu, and Y. He, "PNR2: ranking sentences with positive and negative reinforcement for query-oriented update summarization," in *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1* Manchester, United Kingdom: Association for Computational Linguistics, 2008.

[99] X. C. Jin Zhang, Hongbo Xu, "GSPSummary: A Graph-Based Subtopic Partition Algorithm for Summarization," *Asia Information Retrieval Symposium - AIRS,* pp. 321-334, 2008.

[100] S. Li, Y. Ouyang, W. Wang, and B. Sun, "Multi-document summarization using support vector regression," in *in Proceedings of the Document Understanding Conference* New York, 2007.

[101] R. M. Alguliev, R. M. Aliguliyev, and N. R. Isazade, "Multiple documents summarization based on evolutionary optimization algorithm," *Expert Systems with Applications,* vol. 40, pp. 1675-1689, 2013.

[102] R. M. Alguliev, R. M. Aliguliyev, and C. A. Mehdiyev, "Sentence selection for generic document summarization using an adaptive differential evolution algorithm," *Swarm and Evolutionary Computation,* 2011.

[103] R. M. Alguliev, R. M. Aliguliyev, and M. S. Hajirahimova, "GenDocSum + MCLR: Generic document summarization based on maximum coverage and less redundancy," *Expert Systems with Applications,* vol. 39, pp. 12460-12473, 2012.