UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Revista Científica

http://revistas.udistrital.edu.co/ojs/index.php/revcie/index

ARTÍCULO DE REVISIÓN

Shuffled Frog-Leaping Algorithm Metaheuristic for Extractive Single-Document Summarization

Algoritmo metaheurístico de saltos de ranas mezcladas para la generación automática de resúmenes extractivos de un solo documento

Algoritmo metaheurístico de salto de sapo baralhado para sumarização extrativa de um único documento

Juan-David Yip-Herrera[1]
Martha-Eliana Mendoza-Becerra[2]

Recibido: 14 de agosto de 2024 **Aceptado:** 2 de diciembre de 2024

Para citar este artículo: Yip-Herrera, J. D. y Mendoza-Becerra, M. E. (2024). Shuffled Frog-Leaping Algorithm Metaheuristic for Extractive Single-Document Summarization. *Revista Científica*, *51*(3), 47-61. https://doi.org/10.14483/23448350.22578

Abstract

Due to the increasing amount of information available on the Internet, it is important for users to have a summary containing the most important ideas from the documents they find, in order to quickly identify which ones to read. This article addresses this issue through a modified algorithm for the automatic generation of single-document extractive summaries, aiming to produce summaries of a quality comparable to those generated by expert humans. This proposal is based on the shuffled frog-leaping metaheuristic algorithm (SFLA) and includes a global explicit tabu memory. Its goal is to optimize a weighted objective function with characteristics such as length (measured in words), position within the document, similarity to the document's title, cohesion (similarity between the sentences in the summary), and coverage (similarity between the sentences in the summary and the document). To this effect, an iterative research procedure was followed, consisting of four stages (observation, problem identification, development, and solution testing) over two iterative cycles. In the first cycle, the initialization and evolution schemes were analyzed and selected to modify the base algorithm. This, in addition to parameter tuning. In the second cycle, a tabu memory was selected for integration into the proposed algorithm, and the corresponding tuning was performed. To evaluate the quality of the summaries generated by our proposal, ROUGE metrics were used on the DUC datasets. The results are comparable to and surpass those of various methods in the state of the art. The proposed algorithm stands out for its simplicity of implementation and the reduced number of objective function evaluations, which implies lower computation times.

Keywords: algorithms; artificial intelligence; automatic text analysis; data processing; information retrieval.

1. M.Sc. Universidad del Cauca (Popayán, Cauca, Colombia). juanyip@unicauca.edu.co
2. Ph.D. Universidad del Cauca (Popayán, Cauca, Colombia). mmendoza@unicauca.edu.co

Resumen

Debido a la creciente cantidad de información disponible en Internet, para un usuario es importante contar un resumen con las ideas más importantes delos documentos que encuentra, con el propósito de identificar rápidamente cuáles debe leer. En este artículo se aborda esta problemática mediante un algoritmo modificado para la generación automática de resúmenes extractivos de un solo documento, el cual busca obtener resúmenes de calidad similar a aquellos generados por humanos expertos. Esta propuesta está basada en el algoritmo metaheurístico de saltos de ranas mezcladas e incluye una memoria tabú explícita global. Su propósito es optimizar una función objetivo ponderada con características como longitud (medida en palabras), posición en el documento, similitud con el título del documento, cohesión (similitud entre las oraciones del resumen) y cobertura (similitud entre las oraciones del resumen y el documento). Para ello, se siguió un procedimiento de investigación iterativa compuesto por cuatro etapas (observación, identificación del problema, desarrollo y prueba de la solución) en dos ciclos iterativos. En el primer ciclo se realizó el análisis y selección de los esquemas de inicialización y de evolución, en aras de modificar el algoritmo base. Esto, además del afinamiento de parámetros. Por su parte, en el segundo ciclo se seleccionó una memoria tabú para su integración en el algoritmo propuesto, y se realizó el afinamiento correspondiente. Para evaluar la calidad de los resúmenes obtenidos con nuestra propuesta, se usaron las métricas ROUGE sobre los conjuntos de datos de la DUC. Los resultados se equiparan y superan a diversos métodos del estado del arte. El algoritmo propuesto se diferencia por su simplicidad de implementación y por el reducido número de evaluaciones de la función objetivo, lo que implica un menor tiempo computacional.

Palabras clave: algoritmo; análisis automático de textos; inteligencia artificial; procesamiento de datos; recuperación de información.

Resumo

Devido ao aumento da quantidade de informações disponíveis na Internet, é importante que os usuários tenham um resumo contendo as ideias mais importantes dos documentos que encontram, a fim de identificar rapidamente quais deles ler. Este artigo aborda essa questão através de um algoritmo modificado para a geração automática de sumários extrativos de documento único, visando produzir resumos de qualidade comparável aos gerados por humanos especialistas. Esta proposta é baseada no algoritmo metaheurístico de salto de rã (SFLA) e inclui uma memória tabu global explícita. Seu objetivo é otimizar uma função objetiva ponderada com características como comprimento (medido em palavras), posição dentro do documento, semelhança ao título do documento, coesão (semelhança entre as frases no resumo) e cobertura (semelhança entre as frases no resumo e o documento). Para isso, foi seguido um procedimento de pesquisa iterativo, composto por quatro etapas (observação, identificação de problemas, desenvolvimento e teste de solução) ao longo de dois ciclos iterativos. No primeiro ciclo, os esquemas de inicialização e evolução foram analisados e selecionados para modificar o algoritmo base. Isto, além de ajuste de parâmetros. No segundo ciclo, uma memória tabu foi selecionada para integração ao algoritmo proposto e o ajuste correspondente foi realizado. Para avaliar a qualidade dos resumos gerados por nossa proposta, foram utilizadas as métricas do ROUGE nos conjuntos de dados da DUC. Os resultados são comparáveis e superam os de vários métodos no estado da arte. O algoritmo proposto destaca-se pela sua simplicidade de implementação e pelo reduzido número de avaliações de funções objetivas, o que implica menores tempos de cálculo.

Palavras-chaves: algoritmos; análise automática de texto; inteligência artificial; processamento de dados; recuperação de informação.

INTRODUCTION

The automatic generation of extractive summaries allows characterizing the most relevant information in a document. Its areas of application include e-learning, search engines, e-mail, news, and articles, among others (Yip-Herrera *et al.*, 2023). As shown in El-Kassas *et al.* (2021), multiple methods have been proposed, evaluated, and classified according to *i*) the number of documents, *ii*) the purpose of the summary, *iii*) the target audience, *iv*) the language, and *v*) the extraction method, *i.e.*, **abstractive**, to obtain for summaries with sentences that are not necessarily in the original document (this method uses linguistic analysis tools and focuses on coherence); **extractive**, which involves the specific selection of the most relevant sentences in a document; and **hybrid**, which combines both techniques.

Extractive systems are more common due to their simplicity, computation times, and results quality, and they have been extensively studied; the literature contains a great variety of methods for automatic single document summarization (ASDS) (El-Kassas *et al.*, 2021; Gambhir & Gupta, 2017; Janjanam & Reddy, 2019) based on memetic algorithms, metaheuristics, hybrid particle swarm optimization approaches, graphs, statistics, semantics, differential evolution with self-organizing maps, or clustering (which outperforms metaheuristics).

Metaheuristics-based methods treat summary generation as a global optimization problem, seeking to select the best sentences to form the summary. This approach has obtained satisfactory results (Debnath *et al.*, 2021), positioning itself as an important research area. One specific metaheuristic technique is worth a great deal of attention, *i.e.*, the shuffled frog-leaping algorithm (SFLA) (Bhattacharjee & Sarmah, 2014; Liu *et al.*, 2018), as it combines global and local search (like a memetic algorithm, without including knowledge of the problem). It works like a swarm, is easy to implement, and obtains satisfactory results compared to state-of-the-art methods in optimizing multimodal and non-separable functions as well as solving binary problems such as the knapsack problem, the quadratic knapsack problem, or, for instance, ASDS.

The main contribution of this paper is a SFLA with a global explicit tabu memory (SFLA-ETM) to solve constrained extractive ASDS problems. The modifications made aimed to avoid over-exploitation and find feasible solutions in promising regions. This metaheuristic technique has never been used to solve this problem.

METODOLOGY

As this research project involved a computational solution, we used the iterative field research (IFR) approach proposed by Pratt (2009). This method comprises four main stages, *i.e.*, observation, identification, development, and testing, organized in two cycles involving several iterations. In the first cycle, we studied multiple initializations and evolution schemes (observation), selecting mutation or repair procedures to maintain binary feasible solutions while exploring the search space (identification). Then, we modified the MDSFLA in order to adapt the selected procedures to the ASDS problem (ASDS-SFLA) and adjust the parameters of the algorithm (development). Finally, we evaluated each modification to select the best configuration (testing). After completing the first cycle, we noticed a convergence of solutions, so it was necessary to improve the exploitation and exploration phase. In the second cycle, we studied different tabu memory adaptations (observation), selected the most efficient design (identification), and integrated the tabu memory component into the ASDS-SFLA, thereby obtaining the SFLA-ETM method (development). Lastly, we tested the component with different parameters.

Proposed shuffled frog-leaping algorithm

Characteristics of the objective function

To define the relevance of a candidate summary (S) from a document (D) with a defined number of sentences (ns), an objective function with five weighted characteristics was used, based on the proposals of Mendoza *et al.* (2014, 2015). These characteristics are detailed below.

Sentence position. According to C.-Y. Lin and Hovy (1997), the relevant information of a document is usually found in its titles, its headings, and the first sentences of its paragraphs. This characteristic is calculated via Equation (1), which was taken from Mendoza *et al.* (2015).

$$P(S)=\frac{\sum_{\forall s_i \in S}\frac{2}{n}\left(\frac{n-pos_i}{n-1}\right)}{ns} \tag{1}$$

where n is the total number of sentences in D and pos_i is the position of the sentence s_i.

Relationship between the sentences and the title. This characteristic measures the similarity of s_i to the title of D and assumes that the most similar sentences are the most important. This is calculated via Equation (2) (Mendoza *et al.*, 2014).

$$TRF(S,t)=\frac{\sum_{s_i \in S} cosine\_similarity(s_i,title)}{ns * max_{\forall S}\left[cosine\_similarity(s_i,title)\right]} \tag{2}$$

Sentence length. To estimate the relevance of s_i in the document according to its length in words, Equations (3) and (4) (Mendoza *et al.*, 2014) perform a balanced evaluation, which favors summaries composed of long and average-length sentences.

$$L(S)=\frac{\sum_{\forall s_i \in S}\frac{1-e^{-\alpha_i}}{1+e^{-\alpha_i}}}{ns} \tag{3}$$

$$\alpha_i=\frac{l_i-\mu(l_i)}{std(l_i)} \tag{4}$$

where l_i is the length in words of s_i; $\mu(l_i)$ is the average length; and $std(l_i)$ is the standard deviation of the sentences in S.

Cohesion. This characteristic indicates the degree to which every s_i deals with the same information, measuring the cosine similarity between the vectors representing the sentences in it. It is calculated using Equation (5) (Mendoza *et al.*, 2014).

$$COH(S)=2*\frac{\sum_{i=1}^{ns-1}\sum_{j=i+1}^{ns} cosine\_similarity(s_i,s_j)}{(ns)(ns-1)} \tag{5}$$

Coverage. This characteristic indicates the proportion of information covered by s_i from D, measuring the cosine similarity between the vector representing the sentences in S and the vector of terms of the document, as shown in Equation (6) (Mendoza *et al.*, 2015).

$$COV(S,D) = cosine\_similarity(R(S),D) \tag{6}$$

where $R(S)$ and D are the TF-IDF vectors representing the summary and the whole document, respectively.

With these characteristics, the proposed algorithm seeks to maximize an objective function score for each of the candidate solutions generated through Equation (7), with a summary size limit (8) and a proportionality constraint (9).

$$Max(f(S)) = \alpha P(S) + \beta TRF(S,t) + \gamma L(S) + \delta COH(S) + \rho COV(S,D) \tag{7}$$

$$\sum_{\forall s_i \in S} l_i \leq L \tag{8}$$

$$\alpha + \beta + \gamma + \delta + \rho = 1 \tag{9}$$

where l_i is the length in words of sentence s_i, L is the number of words allowed in the summary, and $\alpha, \beta, \gamma, \delta,$ and ρ are coefficients for weighting each characteristic in the objective function. When a solution does not comply with the length restriction, 0.0 is assigned as its fitness value.

Shuffled frog-leaping algorithm with explicit tabu memory (SFLA-ETM)

The SFLA-ETM method addresses the ASDS problem based on a modified discrete shuffled frog-leaping algorithm (MDSFLA) for solving the 0/1 knapsack problem (Bhattacharjee & Sarmah, 2014). In our method, each summary solution is a frog represented by a binary vector whose size corresponds to the number of sentences in the document (n). Each vector element represents the presence or absence of a document sentence in the candidate summary. A candidate solution is represented as shown in Equation (10).

$$x_i = \left[x_{i,1}, x_{i,2}, \ldots, x_{i,s}, \ldots, x_{i,n} \right], x_i \in P \tag{10}$$

SFLA-ETM is detailed in Figure 1. The modified codes are in bold letters, and the new ones are in bold and shaded. First, a single frog is randomly initialized, set as the best global solution (x_g), and added to the explicit global tabu memory (TM), which works as a first input first output ($FIFO$) list with a defined length, *i.e.*, $Tenure$ (lines 02-04). Then, the frog population (P) is filled (lines 05-12) as follows:

- randomly initialize x_i confirming that it is not in TM
- evaluate its fitness
- add it to TM
- update x_g with x_i if it is a better solution ($x_g \leftarrow x_i$)
- add x_i to P

This process is repeated $N-1$ times.
Afterwards, the evolution process takes place for T iterations (lines 13-23):

- $sortTheFrogs(P)$: sort all frogs in P by their fitness value in descending order
- $shuffleTheFrogs(P, memes)$: shuffle the frogs in P into M memeplexes ($memes$) sequentially, *i.e.*, first frog in the first memeplex, the second frog in the second memeplex, and frog m in the m -th memeplex. Frog $m+1$ goes to the first memeplex, and so on
- Execute ***LocalSearch*** within each memeplex for $maxIter$ iterations in order to improve some solutions (Figure 2)
- $RegroupTheFrogs(memes)$, take out all frogs from the $memes$, place them into P, and order them by fitness value
- $Mutate(x_i)$: each frog attempts to mutate (lines 18-22) by eliminating a random sentence and adding a new one (if possible)
- $Repairsolution(x_i)$: verify whether a solution is feasible according to **(8)**. If it is not, a random sentence is eliminated until it is.

After T iterations or upon reaching the maximum number of objective function evaluations ($FFE \leq 1600$), the algorithm returns the best frog in the pond x_g and constructs the summary for evaluation.

TM : tabu memory list, $memes$: list of M memeplexes, P : population,
N : number of frogs in pond

```
01  Begin
02   Initialize(x_1)                    /* Create the first frog in the pond */
03   x_g ← x_1                          /* Assign frog x_1 as the best global */
04   TM ← TM ∪ {x_1}                    /* Add x_1 to tabu memory */
05   Repeat
06   |  Initialize(x_i)                 /* Create the i-th frog in the pond */
07   |  if x_i exist in TM continue     /* Avoid the use of similar frogs */
08   |    Fitness(x_i)                  /* Evaluate the quality of the i-th frog */
09   |    TM ← TM ∪ {x_i}                /* Add x_i to tabu memory */
10   |    Update(x_g, x_i)               /* If x_i is better than x_g, then x_g ← x_i */
11   |    P ← P ∪ {x_i}                  /* Add x_i to P */
12   Until |P| < N
13   For t ← 1 to T do                  /* Execution of iterations */
14   |  SortTheFrogs(P)                 /* Sort P in descending order */
15   |  ShuffleTheFrogs(P, memes)       /* Distribute frogs into M */
16   |  LocalSearch(memes)              /* Generate new frogs with leaps*/
17   |  P ← RegroupTheFrogs(memes)      /* Collect frogs from M */
18   |  For i ← 2 to N do               /* Apply mutations except x_g */
19   |  |    Mutate(x_i)                /* Mutate i-th frog */
20   |  |    Update(x_g, x_i)           /* If x_i is better than x_g, then x_g ← x_i */
21   |  |    RepairSolution(x_i)        /* Update x_i if necessary */
22   |  Next i
23   Next t                             /* Return the best solution found x_g */
24   Return x_g
25  End
```

Figure 1. *SFLA-ETM procedure*

In the ***LocalSearch*** procedure (Figure 2), the first step is to update x_g if a better solution x_{new} appears (line 04), in addition to identifying the best and worst frogs, *i.e.*, x_b and x_w (line 05), respectively. Then, x_w makes a guided leap (Bhattacharjee & Sarmah, 2014) towards one of the best solutions, trying to improve its fitness value while avoiding recently visited solutions, which are stored in TM . During a guided leap, the components of the frog are processed sentence by sentence ($i = 1, \ldots, n$), and changes are applied only when the values of two frogs are different. The first step is expressed in Equation (11).

$$s_i = rand() * \left(x_{bi} - x_{wi} \right) \tag{11}$$

where s_i is the result of the difference between the best frog (local or global) and the worst frog, multiplied by a uniform random number between [0, 1], this value, which is between [-1, +1], must then be normalized to obtain a binary representation (t_i) using a sigmoidal function. Equation (12) shows this process:

$$t_i = \frac{1}{1 + e^{-s_i}} \tag{12}$$

Next, the definitive value of s_i in x_{new} is calculated via Equation (13), considering the ranges in which t_i is evaluated, as delimited by the parameter α, called *static probability*.

$$x_{newi} = \begin{Bmatrix} 0, & if t_i \leq \alpha \\ x_{wi}, & if\, \alpha < t_i \leq \frac{1}{2}(1+\alpha) \\ 1, & if \frac{1}{2}(1+\alpha) < t_i \end{Bmatrix} \tag{13}$$

First, x_w makes leaps towards x_b (lines 06-08), adds the solution x_{new} to TM , and attempts to replace x_w with x_{new} (lines 06-11). If the solution does not improve, x_w makes a second leap towards x_g (lines 12-17). Finally, if neither of the guided leaps leads to improvement, x_w makes a random leap (lines 18-22). This procedure is repeated for $maxIter$ iterations in the M memeplexes.

```
01  Begin
02    for m = 1 to M do                                   /* For each memeplex */
03    |  for itert = 1 to maxIter do                      /* Maximum iterations in the memeplex */
04    |  |  Update(x_g, x_new)                           /* Define the best frog in the pond */
05    |  |  Calculate(x_w, x_b)                          /* Define the worst and best frogs*/
06    |  |  Repeat
07    |  |  |  x_new ← Leap(x_w, x_b)                   /* First attempt to improve with x_b */
08    |  |  until y doesn't exist in TM
09    |  |  TM ← TM ∪ {x_new}                           /* Add x_new to TM */
10    |  |  Fitness(x_wnew)                             /* Evaluate the quality (fitness) of x_new */
11    |  |  if Update(x_w, x_new) continue              /* Attempt to update x_w */
12    |  |  Repeat
13    |  |  |  x_wnew ← Leap(x_w, x_g)                  /* Second attempt to improve with x_g */
14    |  |  until y does not exist in TM
15    |  |  TM ← TM ∪ {x_new}                           /* Add x_new to TM */
16    |  |  Fitness(x_new)                              /* Evaluate the quality (fitness) of x_new */
17    |  |  if Update(x_w, x_new) continue              /* Attempt to update x_w */
18    |  |  Repeat
19    |  |  |  Initialize(x_w)                          /* Create a new random frog */
20    |  |  until x_w does not exist in TM
21    |  |  TM ← TM ∪ {x_w}                             /* Add x_w into TM */
22    |  |  Fitness(x_w)                                /* Evaluate the quality (fitness) of x_new */
23    |  Next iteration
24    Next m
25  End
```

Figure 2. *LocalSearch procedure*

EXPERIMENTATION AND EVALUATION

To evaluate the proposed algorithm, the 2001 and 2002 datasets of the Document Understanding Conference (DUC) were used, which consist of 309 and 567 news pieces. Each article is accompanied by 100-word reference summaries. The news pieces were processed using natural language processing techniques (El-Kassas *et al.*, 2021) such as segmentation; stopwords, capital letters, and punctuation marks removal; stemming, and indexing.

The ROUGE-*N* metric (version 1.5.5) provided by C. Lin (2004) was used with *N*=1,2. This metric is widely recognized and accepted by the scientific community for comparing the content of automatically generated summaries.

The algorithm was implemented in the C# programming language from the .NET platform, and it was executed on a desktop computer with a 3.4 GHz Intel Core i5 processor, 16 GB RAM, and Windows 10. The full code can be downloaded from https://github.com/juanyip/SDS-MSFLA.git.

To determine the best settings for the proposed algorithm, several configurations were tested, varying the structure of the objective function with different weights and formulas for the characteristics (Table 1); the word inclusion and exclusion criteria (Tables 2 and 3); the repair methods (Table 4); algorithm parameters such as the population (Table 5), the number of memeplexes (Table 6), and the number of total iterations within the memeplexes (Table 7); and the tenure as a TM parameter (Table 8). All tests were executed without exceeding the maximum number of fitness function evaluations (FFEs), *i.e.*, 1600, in order to compare the results against those of other state-of-the-art methods.

Table 1. *Results obtained using different objective functions*

Objective function configuration	DUC2001		DUC2002	
	ROUGE-1	***ROUGE-2***	***ROUGE-1***	***ROUGE-2***
OF_1	***0.45953***	*0.20653*	***0.48738***	***0.22934***
OF_2	*0.43543*	*0.18050*	*0.46598*	*0.20556*
OF_3	*0.45800*	***0.20746***	*0.48429*	*0.22820*
OF_4	*0.44673*	*0.18913*	*0.47370*	*0.21440*
OF_5	*0.45809*	*0.20279*	*0.48634*	*0.22800*

Table 2. *Results obtained by varying the sentence inclusion criteria*

Criterion	DUC2001		DUC2002	
	R1R	***R2R***	***R1R***	***R2R***
Position coverage	*0.45654*	*0.20493*	*0.48824*	*0.23051*
Coverage	*0.45645*	*0.20421*	*0.48710*	*0.22932*
Position	*0.45643*	***0.20592***	***0.48990***	***0.23215***
Cohesion	***0.45729***	*0.20496*	*0.48763*	*0.22895*
Similarity with title	*0.45524*	*0.20232*	*0.48706*	*0.22879*
Sentence length	*0.45471*	*0.20299*	*0.48667*	*0.22937*

Table 3. *Results obtained by varying the sentence exclusion criteria*

Criterion	DUC2001		DUC2002	
	R1R	**R2R**	**R1R**	**R2R**
Position coverage	*0.45481*	*0.20436*	*0.48785*	*0.22908*
Coverage	*0.45402*	*0.20284*	*0.48540*	*0.22627*
Position	*0.45643*	***0.20592***	***0.48990***	***0.23215***
Cohesion	*0.45523*	*0.20255*	*0.48612*	*0.22611*
Similarity with title	***0.45671***	*0.20409*	*0.48418*	*0.22409*
Sentence length	*0.45318*	*0.20291*	*0.48433*	*0.22561*

Table 4. *Results obtained by varying the repair method*

Method	DUC2001		DUC2002	
	R1R	***R2R***	***R1R***	***R2R***
No_repair	*0.45542*	*0.20208*	*0.48288*	*0.22527*
Repair1_Xw	*0.45847*	*0.20618*	*0.48562*	*0.22789*
Repair1_Xw_Mutation	***0.45955***	***0.20690***	*0.48539*	*0.22777*
Repair1_Initialization_Xw	*0.45780*	*0.20552*	*0.48449*	*0.22724*
Repair2_XbXg	*0.45788*	*0.20487*	*0.48530*	*0.22814*
Repair2_XbXgXw	*0.45923*	*0.20576*	*0.48620*	*022882*
Repair2_all	*0.45755*	*0.20450*	***0.48646***	***0.22921***

Table 5. *Results obtained by varying the population*

Population	DUC2001		DUC2002	
	R1R	***R2R***	***R1R***	***R2R***
20	***0.45526***	***0.20320***	***0.48463***	***0.22683***
50	*0.45302*	*0.19942*	*0.48253*	*0.22443*
100	*0.45111*	*0.19882*	*0.48036*	*0.22189*
150	*0.45263*	*0.20073*	*0.47937*	*0.22229*
200	*0.44903*	*0.19827*	*0.47805*	*0.22053*

Table 6. *Results obtained by varying the number of memeplexes*

No. Memeplexes	DUC2001		DUC2002	
	R1R	***R2R***	***R1R***	***R2R***
2	*0.44703*	*0.19508*	*0.48373*	*0.22676*
4	*0.45196*	*0.19629*	*0.48330*	*0.22527*
5	***0.45596***	***0.20324***	*0.48065*	*0.22394*
10	*0.45526*	*0.20320*	***0.48463***	***0.22683***
20	*0.44507*	*0.19437*	*0.47782*	*0.22133*

Table 7. *Results obtained by varying the iterations within the memeplex*

Memeplex iterations	DUC2001		DUC2002	
	R1R	***R2R***	***R1R***	***R2R***
5	*0.45474*	*0.20231*	*0.48458*	*0.22682*
10	*0.45526*	*0.20320*	***0.48463***	*0.22683*
20	***0.45682***	***0.20401***	*0.48011*	*0.22352*
30	*0.45460*	*0.20313*	*0.48377*	*0.22568*
40	*0.45365*	*0.19880*	*0.48348*	***0.22725***

Table 8. *Explicit global tenure tunning results*

Explicit global tenure	DUC2001		DUC2002	
	R1R	***R2R***	***R1R***	***R2R***
8	***0.45746***	***0.20678***	***0.49140***	***0.23214***
9	*0.45717*	*0.20654*	*0.49005*	*0.23186*
10	*0.45728*	*0.20639*	*0.49029*	*0.23207*

To tune the objective function, a range of weight combinations was generated for the characteristics, with the best results being as follows: $P = 0.15, TRF = 0.04, L = 0.09, COH = 0.07, and\ COV = 0.65$. The parameter values were tested based on problem knowledge and their effect on the exploration and exploitation process, obtaining the following results: $P = 20$, $M = 5$, $prMut = 0.06$, $MaxIter = 10$, $\propto = 0.4$, and $Tenure = 8$.

RESULTS AND DISCUSSION

The quality of the summaries obtained by the proposed algorithm was compared against those generated through state-of-the-art methods based on individual or combined approaches, *i.e.*, metaheuristic, evolutionary, graph-based, and memetic algorithms including clustering or multi-objective techniques. All values obtained and taken from the literature are presented in Table 9. The best results found are underlined, and those of SFLA-ETM are in bold. Our approach ranked seventh in ROUGE-1 for DUC01, third on ROUGE-1 for DUC02, and fourth in the other instances.

Table 9. *Results for DUC01/DUC02 with ROUGE-1 and ROUGE-2*

Method	Year	DUC 2001		DUC 2002	
		ROUGE-1	ROUGE-2	ROUGE-1	ROUGE-2
MBCSO (Debnath *et al.*, 2023)	2023	0.65124 (1)	0.32280 (1)	0.67333 (1)	0.34985 (1)
EDGESUM (El-Kassas *et al.*, 2020)	2020	0.51374 (2)	0.27166 (2)	0.53795 (2)	0.28584 (3)
SFLA-ETM	**2023**	**0.45746 (7)**	**0.20678 (4)**	**0.49140 (3)**	**0.23214 (4)**
ESDS_SMODE (Saini *et al.*, 2019)	2019	0.45214 (11)	0.21450 (3)	0.49117 (4)	0.34132 (2)
COSUM (Alguliyev *et al.*, 2019)	2019	0.47270 (5)	0.20120 (6)	0.49080 (5)	0.23090 (5)
MA-SingleDocSum (Mendoza *et al.*, 2014)	2014	0,44862 (13)	0.20142 (5)	0.48280 (8)	0.22840 (7)
ETS-GA	2018	0.45058 (12)	0.19619 (8)	0.48423 (7)	0.22471 (8)
ESDS-GHS-GLO (Mendoza *et al.*, 2015)	2015	0.45402 (9)	0.19565 (9)	0.47896 (10)	0.22138 (9)
LexRank (Erkan & Radev, 2004)	2004	0.44680 (15)	0.19890 (7)	0.47960 (9)	0.22950 (6)
UnifiedRank (Wan, 2010)	2010	0,45377 (10)	0.17646 (14)	0.48487 (6)	0.21462 (10)
FEOM (Song *et al.*, 2011)	2006	0.47728 (4)	0.18549 (10)	0.46575 (13)	0.12490 (15)
DE (Aliguliyev, 2009)	2009	0.47856 (3)	0.18528 (11)	0.46694 (12)	0.12368 (16)
NetSum (Svore *et al.*, 2007)	2007	0.46427 (6)	0.17697 (13)	0.44963 (14)	0.11167 (17)
CollabSum (Wan *et al.*, 2007)	2007	0.44040 (16)	0.16230 (16)	0.47190 (11)	0.20100 (11)
QSC (Dunlavy *et al.*, 2007)	2007	0.44852 (14)	0.18523 (12)	0.44865 (15)	0.18766 (14)
CRF (Shen *et al.*, 2007)	2007	0.45512 (8)	0.17327 (15)	0.44006 (16)	0.10924 (18)
ESDS_MGWO (Saini *et al.*, 2019)	2019	0.37108 (17)	0.15228 (17)	0.41849 (17)	0.18838 (12)
ESDS_MWCA (Saini *et al.*, 2019)	2019	0.36702 (18)	0.14997 (18)	0.41800 (18)	0.18812 (13)

As seen in Table 9, it was not easy to define the best algorithm. Therefore, we used a unified ranking, considering the position occupied by each method with regard to each measure. To his effect, Equation (14) was used (Mendoza *et al.*, 2014).

$$Rank\left(Method\right) = \sum_{r=1}^{TNM} \frac{\left(TNM - r + 1\right) R_r}{TNM} \tag{14}$$

where R_r indicates the number of times that the method ranked in position r, and TNM is the total number of methods being compared ($TNM = 18$). The unified rankings are presented in Table 10, where higher values are better.

Table 10. *Algorithm ranking*

#	Method	*Rr*																		Rank value
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	
1	MBCSO	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4.00
2	EDGESUM	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3.72
3	**SFLA-ETM**	**0**	**0**	**1**	**2**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**3.22**
4	ESDS_SMODE	0	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	3.11
5	COSUM	0	0	0	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0	3.06
6	MA-SingleDocSum	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	2.39
7	ETS-GA	0	0	0	0	0	0	1	2	0	0	0	1	0	0	0	0	0	0	2.28
8	ESDS-GHS-GLO	0	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	2.17
9	LexRank	0	0	0	0	0	1	1	0	1	0	0	0	0	0	1	0	0	0	2.17
10	UnifiedRank	0	0	0	0	0	1	0	0	0	2	0	0	0	1	0	0	0	0	2.00
11	FEOM	0	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0	0	0	1.89
11	DE	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	1.89
13	NetSum	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	1	0	1.44
14	CollabSum	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	2	0	0	1.22
15	QSC	0	0	0	0	0	0	0	0	0	0	0	1	0	2	1	0	0	0	1.17
16	CRF	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	1.06
17	ESDS_MGWO	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	3	0	0.72
18	ESDS_MWCA	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	3	0.50

SFLA-ETM outperformed multiple algorithms, but the best results were obtained by MBCSO, which outperformed all other algorithms in all metrics, followed by EDGESUMM. MBCSO stands for multi-objective binary cat swarm optimization, and it uses modified methods to avoid getting stuck in local optima. However, it requires approximately 6000 evaluations of the objective function. On the other hand, EDGESUMM is a graph-based framework that combines four known algorithms (graph-, statistics-, semantics-, and centrality-based), albeit exhibiting all their disadvantages as well as increased complexity.

In Table 10, the proposed method ranks third, with competitive results. It outperforms graph-based algorithms, neural networks, differential evolution techniques, and other metaheuristic approaches. As with MBCSO, our proposal uses short-term memory, albeit with much fewer FFEs required (1600) due to its fast convergence. Therefore, our algorithm is much more efficient in terms of computation times. MBCSO has other disadvantages related to the large number of characteristics it uses (7) and its increased complexity, which stems from the calculation of two characteristics in the objective function (*i.e.*, similarity with the title and antiredundancy) using word-mover distance (WMD). WMD is a method that calculates the similarity and dissimilarity between sentences, calculating the distance between them via *word2vec*. This is done based on their meaning, which is useful when there are no common words (*i.e.*, semantic similarity). On the other hand, EDGESUMM uses four algorithms to process the document, starting with semantic clustering, which implies more complexity. In contrast, our proposal has the advantage that it uses a single metaheuristics-based algorithm, so it is much more efficient in terms of computation times (which are even lower than the first two algorithms in the ranking) and memory usage. This is useful in applications requiring real-time response, such as emergency response summarization to aid decision-making.

CONCLUSIONS

The proposed SFLA-ETM is an adaptation from Bhattacharjee and Sarmah (2014) to solve the ASDS problem with a single objective function that integrates a global explicit tabu memory. This method outperformed other graph-based algorithms, metaheuristics methods, genetic algorithms, differential evolution algorithms, particle swarm optimizers, and hybrids techniques, as well as other approaches that combine several algorithms, which are more complex, require more evaluations, and exhibit increased processing times. The main modifications made involved the management of non-feasible solutions during initialization, the inclusion of repair procedures in the evolution process, a multi-bit mutation process that improves exploitation, and the adaptation of an overall tabu memory that allows for local search, thereby increasing diversity and avoiding the unnecessary evaluation of recently visited candidate solutions.

Although most research works on ASDS indicate that title relationship is important, when tuning the weights of the objective function, coverage exhibited a relevance of 65, which may be due to the fact that this characteristic searches for sentences that are part of the summary while considering the entirety of the document.

This research contributes to the local and regional scientific community dedicated to automatic single-document summarization with the knowledge and algorithm presented in this paper. This algorithm could be applied to similar problems, *e.g.*, shipment packing problems, which are modeled just as the knapsack problem.

In future work, we believe it is necessary: *i*) to study other methods of initialization, repair, and replacement; *ii*) to review other mathematical expressions for the characteristics of the objective function, aiming to make summary evaluations closer to those performed by humans; and *iii*) to study the behavior of the objective function with other datasets; *iv*) to adapt other metaheuristic algorithms to the ASDS problem, which should combine global and local search and should not require large numbers of evolutionary operators; and *v*) to use different multi-objective evolutionary algorithms such as NSGA-III and MOEA/D in the studied problem.

ACKNOWLEDGEMENTS

We are especially grateful to Universidad del Cauca for their support, as well as to Mr. Colin McLachlan for reviewing this document.

AUTHOR CONTRIBUTIONS

Juan-David Yip-Herrera: investigation, software, validation, writing (original draft, review, & editing).
Martha-Eliana Mendoza: conceptualization, supervision, investigation, writing (review & editing).

REFERENCES

Alguliyev, R. M., Aliguliyev, R. M., Isazade, N. R., Abdi, A., Idris, N. (2019). COSUM: Text summarization based on clustering and optimization. *Expert Systems*, *36*(1), 1-17. https://doi.org/10.1111/exsy.12340

Aliguliyev, R. M. (2009). A new sentence similarity measure and sentence based extractive technique for automatic text summarization. *Expert Systems with Applications*, *36*(4), 7764-7772. https://doi.org/10.1016/j.eswa.2008.11.022

Bhattacharjee, K. K. K., Sarmah, S. P. P. (2014). Shuffled frog leaping algorithm and its application to 0/1 knapsack problem. *Applied Soft Computing*, *19*, 252-263. https://doi.org/https://doi.org/10.1016/j.asoc.2014.02.010

Debnath, D., Das, R., Pakray, P. (2021). Extractive single document summarization using multi-objective modified cat swarm optimization approach: ESDS-MCSO. *Neural Computing and Applications*, *4*, s00521-021-06337-4. https://doi.org/10.1007/s00521-021-06337-4

Debnath, D., Das, R., Pakray, P. (2023). Single document text summarization addressed with a cat swarm optimization approach. *Applied Intelligence*, *53*(10), 12268-12287. https://doi.org/10.1007/s10489-022-04149-0

Dunlavy, D. M., O'Leary, D. P., Conroy, J. M., Schlesinger, J. D. (2007). QCS: A system for querying, clustering and summarizing documents. *Information Processing and Management*, *43*(6), 1588-1605. https://doi.org/10.1016/j.ipm.2007.01.003

El-Kassas, W. S., Salama, C. R., Rafea, A. A., Mohamed, H. K. (2020). EdgeSumm: Graph-based framework for automatic text summarization. *Information Processing & Management*, *57*(6), e102264. https://doi.org/10.1016/j.ipm.2020.102264

El-Kassas, W. S., Salama, C. R., Rafea, A. A., Mohamed, H. K. (2021). Automatic text summarization: A comprehensive survey. *Expert Systems with Applications, 165*, e113679. https://doi.org/10.1016/j.eswa.2020.113679

Erkan, G., Radev, D. R. (2004). LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, *22*(4), 457-479. https://doi.org/10.1613/jair.1523

Gambhir, M., Gupta, V. (2017). Recent automatic text summarization techniques: A survey. *Artificial Intelligence Review*, *47*(1), 1-66. https://doi.org/10.1007/s10462-016-9475-9

Janjanam, P., Reddy, C. H. P. (2019). *Text summarization: An essential study* [Conference paper]. 2nd International Conference on Computational Intelligence in Data Science. https://doi.org/10.1109/ICCIDS.2019.8862030

Lin, C. (2004). Rouge: A package for automatic evaluation of summaries. *Proceedings of the Workshop on Text Summarization Branches out (WAS 2004)*, *1*, 25-26.

Lin, C.-Y., Hovy, E. (1997). *Identifying topics by position* [Conference paper]. Fifth Conference on Applied Natural Language Processing, San Francisco, CA, USA.

Liu, C., Niu, P., Li, G., Ma, Y., Zhang, W., Chen, K. (2018). Enhanced shuffled frog-leaping algorithm for solving numerical function optimization problems. *Journal of Intelligent Manufacturing*, *29*(5), 1133-1153. https://doi.org/10.1007/s10845-015-1164-z

Mendoza, M., Bonilla, S., Noguera, C., Cobos, C., León, E. (2014). Extractive single-document summarization based on genetic operators and guided local search. *Expert Systems with Applications*, *41*(9), 4158-4169. https://doi.org/10.1016/j.eswa.2013.12.042

Mendoza, M., Cobos, C., León, E. (2015). Extractive single-document summarization based on global-best harmony search and a greedy local optimizer. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9414*, 52-66. https://doi.org/10.1007/978-3-319-27101-9_4

Pratt, K. (2009). *Design patterns for research methods: Iterative field research*. Association for the Advancement of Artificial Intelligence.

Saini, N., Saha, S., Jangra, A., Bhattacharyya, P. (2019). Extractive single document summarization using multi-objective optimization: Exploring self-organized differential evolution, grey wolf optimizer and water cycle algorithm. *Knowledge-Based Systems*, *164*, 45-67. https://doi.org/10.1016/j.knosys.2018.10.021

Shen, D., Sun, J.-T., Li, H., Yang, Q., Chen, Z. (2007). *Document summarization using conditional random fields* [Conference paper]. 20th International Joint Conference on Artifical Intelligence.

Song, W., Cheon Choi, L., Cheol Park, S., Feng Ding, X. (2011). Fuzzy evolutionary optimization modeling and its applications to unsupervised categorization and extractive summarization. *Expert Systems with Applications*, *38*(8), 9112-9121. https://doi.org/10.1016/j.eswa.2010.12.102

Svore, K., Vanderwende, L., Burges, C. (2007). *Enhancing single-document summarization by combining RankNet and third-party sources* [Conference paper]. Conference on Empirical Methods in Natural Language Processing (EMNLP-CoNLL).

Wan, X. (2010). *Towards a unified approach to simultaneous single-document and multi-document summarizations* [Conference paper]. 23rd International Conference on Computational Linguistics (pp. 1137-1145).

Wan, X., Yang, J., Xiao, J. (2007). *CollabSum: Exploiting multiple document clustering for collaborative single document summarizations* [Conference paper]. 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'07. https://doi.org/10.1145/1277741.1277768

Yip-Herrera, J.-D., Mendoza-Becerra, M.-E., Rodríguez, F. (2023). Generación automática de resúmenes extractivos para un solo documento: un mapeo sistemático. *Revista Facultad de Ingeniería*, *32*(63), e15232. https://doi.org/10.19053/01211129.v32.n63.2023.15232