A New Memetic Algorithm for Multi-document Summarization Based on CHC Algorithm and Greedy Search

Martha Mendoza[1], Carlos Cobos[1], Elizabeth León[2], Manuel Lozano[3], Francisco Rodríguez[3], and Enrique Herrera-Viedma[3]

[1] Universidad del Cauca, Popayán, Colombia
{mmendoza,ccobos}@unicauca.edu.co
[2] Universidad Nacional de Colombia, Bogotá D.C., Colombia
eleonguz@unal.edu.co
[3] Universidad de Granada, Granada, España
{lozano,fjrodriguez,viedma}@decsai.ugr.es

Abstract. Multi-document summarization has been used for extracting the most relevant sentences from a set of documents, allowing the user to more quickly address the content thereof. This paper addresses the generation of extractive summaries from multiple documents as a binary optimization problem and proposes a method, based on CHC evolutionary algorithm and greedy search, called MA-MultiSumm, in which objective function optimizes the lineal combination of coverage and redundancy factors. MA-MultiSumm was compared with other state-of-the-art methods using ROUGE measures. The results showed that MA-MultiSumm outperforms all methods on the DUC2005 dataset; and on DUC2006 the results are very close to the best method. Furthermore in a unified ranking MA-MultiSumm only was improved on by the DESAMC+DocSum method, which requires as many iterations of the evolutionary process as MA-MultiSumm. The experimental results show that the optimization-based approach for multiple document summarization is truly a promising research direction.

Keywords: Multi-document summarization, Memetic algorithms, CHC algorithm, Greedy search.

1 Introduction

Currently vast quantities of information are found in digital text documents on the internet and within organizations. When a user is interested in exploring a specific topic in depth, the information required may be contained in a large number of related texts that can be read in their entirety by the user only with great difficulty, the user having to invest much time and effort to find what they are looking for; it is therefore important to be able to rely on a summary in order to identify the main topics contained in the documents available. For many years, the automatic generation of summaries has been attempting to create summaries that closely approximate those generated by humans

A. Gelbukh et al. (Eds.): MICAI 2014, Part I, LNAI 8856, pp. 125–138, 2014.

[1, 2], enabling the user to engage the documents that satisfy their requirements in less time. Some of the application areas of the generation of extractive summaries from multiple documents are the summaries of news, web Summarization and email thread summarization [2].

Different taxonomies for summaries exist [1, 2], based on the way the summary is generated, the target audience of the summary, the number of documents to be summarized, and so on. According to the way in which the summary is generated, it can be either extractive or abstractive [1, 2]. Extractive summaries are formed from the reuse of portions of the original text. Abstractive summaries [3], on the other hand, are rather more complex, requiring linguistic analysis tools to construct new sentences from those previously extracted. Taking account of the target audience, summaries may be [1, 2]: generic, query-based, user-focused or topic-focused. Generic summaries do not depend on the audience for whom the summary is intended. Query-based summaries respond to a query made by the user. User-focused ones generate summaries to tailor the interests of a particular user, while topic-focused summaries emphasize those summaries on specific topics of documents. Depending to the number of documents that are processed, summaries [1, 2] can be either single document or multiple document. With regard the language of the document, they may be monolingual or multilingual, and regarding document genre may be scientific article, news, blogs, and so on.

The summarization algorithm (method) proposed in this paper is extractive; generic and multiple documents, allowing the summary is generated on any group of documents; and for any type of document, although the evaluation was performed on a set of news.

Automatic summarization is an area that has explored different methods for the automatic generation of summaries of multiple documents, such as: (1) machine learning technique approaches [4, 5] using training data to identify the characteristics that have the greatest impact on the selection of the sentences that make up the summary; (2) approaches based on text connectivity [6], with lexical strings using lexical databases such as WordNet to find relationships between different words. The chains are classified by their length and homogeneity, and the strongest lexical strings are selected. After each of these chains, sentences are selected to create the summary. Most recently the focus of rhetorical roles has been employed [7]; (3) graph-based approaches [8-10], which represent units of text (key words or sentences) in the vertices of the graph, and the similarity between the text units by means of the edges, then an iterative process is carried out and the summary with sentences of the first vertices is obtained; (4) based on algebraic reduction [11, 12] through LSA (Latent Semantic Analysis) and NMF (Non-negative Matrix Factorization), which make use of matrix decomposition to find the sentences that best represent the document; (5) based on clustering and probabilistic models [13, 14], in which the priority is to generate sets of documents or sentences associated with a particular topic; and (6) based on metaheuristics that seek to optimize an objective function to find the sentences that will be part of the summary [15-17].

Of these methods, those based on algebraic reduction, clustering, probabilistic models and metaheuristics are language independent and unsupervised, aspects on which more emphasis is being placed in recent research so as to avoid dependence on language and training groups. Although these methods have achieved good results over other methods, recent research based on memetic algorithm for single document [18], have shown better results, making research in this area promising, and leaving the possibility of exploring the application of the memetic algorithms for multiple documents that are not currently being used. Further, memetic algorithms have contributed in solving problems of discrete combinatorial optimization obtaining very good results [19]; nevertheless, they have not yet been used to solve the problem of automatic generation of summaries from multiple documents.

In this paper a memetic algorithm based on CHC (Cross-generational elitist selection, Heterogeneous recombination, Cataclysmic mutation) algorithm and greedy search (local search) is proposed, for automatic generation of extractive and generic summaries from multiples documents, in which objective function is optimized by the lineal combination of two factors: coverage that exists between all candidate sentences in the summary and the document collection sentences; and redundancy that exists between the sentences in the summary.

The rest of the paper is organized as follows: Section 2 introduces the document representation and characteristics of the objective function proposed in the algorithm. Section 3 describes the proposed algorithm; while the results of evaluation using data sets, along with a comparison and analysis with other state-of-the-art methods, are presented in Section 4; finally, Section 5 presents conclusions and future work.

2 Problem Statement and Its Mathematical Formulation

The representation of a document is made based on the vector space model proposed by Salton [20]. Thus, a document is represented by the sentences that compose it, in this case, it is represented as the set of all the sentences that the collection of documents contains, i.e. $D=\{S_1, S_2, \ldots, S_n\}$, where S_i corresponds to the i-th sentence of the document collection and n is the total number of sentences in this collection. Likewise, a sentence is represented by the set $S_i = \{t_{i1}, t_{i2}, \ldots, t_{ik}, \ldots, t_{io}\}$, where t_{ik} is the k-th term of the sentence S_i and o is the total number of terms in the sentence. Thus, the vector representation of a sentence of the document is a vector containing the weights of the terms, as shown in Eq. (1).

$$s_i = \{w_{i1}, w_{i2}, \ldots, w_{ik}, \ldots, w_{im}\} \tag{1}$$

Where m is the number of distinct terms in the document collection, w_{i1} is the weight of term t_1 in sentence S_i and w_{ik} is the weight of term t_k in sentence S_i.

The component w_{ik} is calculated as the relative frequency of the term in the document [20]. The scheme assigns the weight as shown in Eq. (2).

$$w_{i,k} = (f_{i,k} / MaxFreq) \times \log(n/(1+n_t)) \tag{2}$$

Where $f_{i,k}$ represents the frequency of term k in sentence S_i, $MaxFreq_i$ is an adjustment factor that indicates the number of occurrences of the most frequent term in the sentence S_i, n_k denotes the number of sentences in which the term t_k appears, and n is the number of sentences in the document collection.

Thus the aim of generating a summary of multiple documents is to obtain a subset of D with the sentences that contain the main information of the document collection. To do this, features are used whose purpose is to evaluate the subset of sentences to determine the extent to which they cover the most relevant information of the document collection. These features are based on measures of similarity between sentences. The similarity between two sentences S_i and S_j, according to the vector representation described, is measured in the same way as the cosine similarity [20] which relates to the angle of the vectors S_i and S_j.

In a memetic algorithm, the objective function is in charge of guide the search of the best summaries based on sentences features. In this paper an objective function based on maximum coverage and minimum redundancy is introduced, taking into account that research that includes these factors in the objective function have shown good results in relation to the state of the art methods [15, 21].

Coverage Factor: A summary ought to contain the main aspects of the documents with the least loss of information. The sentences selected should therefore cover the largest amount of information contained within the set of sentences in the document collection. As such, coverage factor is calculated taking into account the cosine similarity between the text of the candidate summary and the sentences of the entire collection of documents as shown in Eq. (3).

$$Fc = sim_{\cos}(R, D) \tag{3}$$

Where R represents the text with all the candidate summary sentences; D represents all the sentences of the document collection (in this case, it is the centroid of the collection). This factor therefore takes values between zero and one.

Redundancy Factor: Managing redundancy is a very important factor, because the generated summary should avoid containing repeated information in it, that is, have the least redundancy as possible, especially when dealing with the problem of generating summaries of multiple documents covering the same topic. To eliminate redundancy in the sentences of the summary, this factor is calculated based on what was stated in [15], but carrying out a normalization so that this factor takes values between zero and one, as with the coverage factor (See Eq. (4)).

$$Fr = \frac{2}{r \times (r-1)} \sum_{i=1}^{r-1} \sum_{j=i+1}^{r} sim_{\cos}(S_i, S_j) \tag{4}$$

Where S_i and S_j are sentences in the candidate summary and r is the number of sentences in the summary.

Thus the objective function to maximize is defined as the linear combination of the coverage (Fc) and redundancy (Fr) factors (See Eq. (5)). The latter is subtracted in the equation to prevent the generated summary containing identical or similar sentences. A lambda coefficient (λ) is introduced, which gives flexibility to the objective function allowing more or less weight to be given to each factor. The coefficient λ varies between zero and one. Eq. (6) includes a restriction to maximize the information included in the summary by selecting sentences containing relevant information but few words.

Maximize

$$f(x) = \lambda Fc - (1-\lambda)Fr = \lambda(sim_{\cos}(R,D)) - (1-\lambda)\frac{2}{r\times(r-1)}\sum_{i=1}^{r-1}\sum_{j=i+1}^{r} sim_{\cos}(S_i,S_j) \tag{5}$$

subject to

$$\sum_{i=1}^{r} l_i x_i \leq L \tag{6}$$

Where x_i, indicates one if the sentence S_i is selected and zero otherwise; l_i is the length of the sentence S_i (measured in words) and L is the maximum number of words allowed in the generated summary.

3 The Proposed Memetic Algorithm: MA-MultiSumm

In Fig. 1, the general outline of the proposed memetic algorithm for automatically generating extractive summaries based on CHC [22] and greedy search, MA-MultiSumm, is shown. The most important modifications as regarding to the original CHC algorithm, are: (1) the initial value of d is smaller (d_o=0.025×L) than CHC original (0.25×L), because the agent is represented in this problem by many zeros and few ones, therefore the agents are very similar to each other; (2) local search is applied to the agents to find local optimal; and (3) in the cataclysm, the two best individuals are preserved, the remaining individuals are created randomly, and threshold d takes the initial value d_o. In section 3.1 is described the local search strategy used in MA-MultiSumm algorithm.

Population initialization. The initial population is composed of p agents, generated randomly taking into the constraint of the maximum number of words allowed in the summary (the number of sentences in the agent is controlled by means Eq. (6)). Each agent represents the presence of the sentence in the summary by means of a one, and absence with a zero. The most common strategy for initializing the population (t=0) is to randomly generate each agent. So that all the sentences in the document have the same probability of being part of the agent, a random number between one and n (number of sentences in the document collection) is defined, the gene corresponding to this value is chosen and a value of one is given, so that this sentence will become part of the summary in the current agent. Thus, the c-th agent of the initial population is created as shown in Eq. (7).

$$X_c(0) = [x_{c,1}(0), x_{c,2}(0), \ldots, x_{c,n}(0)], \quad x_{c,s}(0) = a_s \tag{7}$$

Where a_s is a random value in {0,1}, c=1,2,…,p and s=1,2,…,n., p is the population size and n is the number of sentences.

Evaluation and optimization of the initial population. After generating the initial population randomly, the fitness value of each agent is calculated using Eqs. (5)-(6). Then a percentage op of the population is optimized using greedy local search, which will be explained later. Finally the fitness is recalculated, and the resulting population is ordered from highest to lowest based on this new fitness value.

```
L: agent length;  p: population size;  d: difference threshold;  op: optimization probability;
dh: hamming distance;  nofe: number of objective function evaluations;
mnofe: maximum number of objective function evaluations;
-----------------------------------------------------------------------------------------------
t = 0;
d = d_o                          // Minimum of different genes (sentences), the value of d_o is 0.025×L.
Initialize (P(t));               // Random initialization, each gen represents the absence or presence
                                 // of the sentence on the summary.
Evaluate (P(t));                 // Calculate fitness for each agent in the population P(t).
Optimization (P(t));             // Only a percentage of P(t) is optimized.
While nofe < mnofe do
    For i= 1… p/2 do
        Selection (p1, p2, P(t));          // Select parent1 (p1) and parent2 (p2) from current population.
        If (dh (p1, p2) < d) Then Continue; // Incest prevention mechanism using the hamming distance.
        HUX_Crossover (p1, p2);            // HUX Crossover among p1 and p2 to obtain offspring.
        For each offspring do              // With two offspring that were created.
            Evaluate (offspring);          // Calculate fitness for the offspring.
            If (U(0,1) < op) Optimization (offspring);   // Only a percentage op of the
                                                         // current population is optimized.
            P(t+1)=Add(offspring);         // Add offspring to the new population.
        End For each;
    End For;
    If (P(t+1) = empty) Then d = d – 1;    // It permits great similarity among the parents.
    P(t+1) = P(t+1) ∪ P(t);                // Merge the members of the current population
                                           // with the generated offspring.
    Preserve best agents from P(t+1);      // When parent and offspring have the same fitness value,
                                           // the offspring is selected.
    If (d = 0) Cataclysm();                // The two best individuals are preserved and the remaining
                                           // individuals are generated randomly.
    t = t +1;
End while;
Return (BestAgent);   // The agent with best fitness in last population is returned;
```

Fig. 1. Scheme of the MA-MultiSumm memetic algorithm

Selection. The generation step starts with the selection operator and is repeated $p/2$ times. The two parent agents are selected randomly from the current population ensuring that they are not repeated.

Incest prevention. This mechanism calculates the Hamming distance between the two parent agents to validate that the total number of distinct genes among them is greater than a threshold d (minimum allowable different genes) and thus avoid incest. If this threshold is not met, new parents are selected.

HUX crossover. To produce the two offspring, HUX crossover strategy is used between the two parents selected. Thus, the genes found in both parents will also part of the offspring and half of the genes that are not equal are exchanged randomly.

Optimization the offspring. A uniform random number between zero and one is generated. If this value is less than the probability of optimization (op), the offspring generated by HUX crossing is optimized using a greedy local search operator. If the fitness value of the optimized agent is better than the fitness value of the agent without optimization, the current agent is replaced by the optimized agent.

Replacement. If in the new generation there are no offspring, the value of *d* is decreased to allow the agents selected as parents to become more similar and generate offspring. Replacement is carried out when the population of agents generated is already full, joining with the current population, which has been sorted previously according to the fitness value. Then, the new population is formed with the *p* best agents from the union of the two populations, giving priority to the offspring when they have fitness equal to that of the parents.

Cataclysm. On generating a new population, whether or not cataclysm occurs in the population is evaluated. For this, whether the minimum number of different genes to prevent incest is less than or equal to zero is checked. When cataclysm occurs, the two agents with the highest fitness value of the current generation are kept and the remaining agents are generated completely randomly according to the process explained in the generation of the initial population.

Stopping criterion. The running of the memetic algorithm terminates when the stop condition is met. The stop condition was established earlier as a maximum number of evaluations of the objective function.

3.1 Greedy Search

Regarding local search, MA-MultiSumm uses a Greedy approach [23], taking into account *op*. The agent is optimized a defined number of times (*Maxnumop*), adding and removing a sentence from the summary, and controlling the number of sentences in the agent by means Eq. (6). If the fitness value of the new agent improves previous agent, the replacement is made. Otherwise, the previous agent is retained. A movement is then made again in the neighborhood, repeating the previous steps (Fig. 2).

```
Lss: a list of sentences sorted for similarity with the documents collection;
Maxnumop: maximum number of optimizations;  OriginalAgent: original agent (agent to optimize);
---------------------------------------------------------------------------------------------
For i=1 … Maxnumop do
    CurrentAgent = Copy (OriginalAgent);
    Add_sentence (CurrentAgent);          // A sentence with the highest value of similarity of
                                          // the list Lss is activated in the agent.
    Delete_sentence (CurrentAgent);       // A sentence with the lowest value of similarity of
                                          // the list Lss is turned off in the agent.
    Length_restriction (CurrentAgent);    // The restriction of the summary length is executed.
    Evaluate (CurrentAgent);              // Calculate fitness for current agent.
    If (Fitness(CurrentAgent) > Fitness(OriginalAgent)) Then OriginalAgent = CurrentAgent;
End For
```

Fig. 2. Procedure of greedy search

The neighborhood is generated based on a scheme of elitism, in which the sentence that is placed in one (i.e. included in the candidate summary) is selected from a list sorted according to the similarity of the sentence to the entire document collection; and the sentence that is placed in zero (thereby being removed from the candidate summary) is the one with least similarity to the entire document collection. This means that the coverage factor is the criterion used to include or remove a sentence from the candidate summary.

4 Experiment and Evaluation

To evaluate the MA-MultiSumm algorithm, Document Understanding Conference (DUC) datasets for the years 2005 and 2006 were used. The DUC2005 collection is comprised of fifty topics, each containing between 25 and 50 documents; and the DUC2006 comprises fifty topics, each with 25 documents. Furthermore the summary generated should be less than 250 words and have several reference summaries for each topic. For each topic the algorithm was run thirty (30) times to obtain the average of each measure for each data set.

Pre-processing of the documents involves linguistic techniques such as segmentation of sentences or words [20], removal of stop words, removal of capital letters and punctuation marks, stemming and indexing [20]. This process is carried out before starting to run the algorithm for the automatic generation of multiple documents.

The segmentation process was done using an open source segmentation tool called "splitta" (available at http://code.google.com/p/splitta). The stop words removal process was done based on the list built for the SMART information retrieval system (ftp://ftp.cs.cornell.edu/pub/smart/english.stop). The Porter algorithm was used for the stemming process. Finally, Lucene (http://lucene.apache.org) was used to facilitate the entire indexing and searching in information retrieval tasks.

Evaluation of the quality of the summaries generated by the MA-MultiSumm method was performed using metrics provided by the assessment tool ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [24], version 1.5.5 (available on internet), which has been widely handled by DUC in evaluating automatic summaries. ROUGE is accepted by DUC as the official metric for the evaluation of automatic summarization of texts.

The comparison of the proposed algorithm was made against DESAMC+DocSum [16], PLSA [13], LFIPP [25], MCMR [15], HybHSum [26], LEX [27], SVR [5], iRANK [28], HierSum [29], Centroid [10], SNMF +SLSS [30], TMR [31], and MMR [32].

4.1 Parameter Tuning

Parameter tuning was carried out based on the Meta Evolutionary Algorithm (Meta-EA) [33], using a version of harmony search [34]. The configuration of parameters for the MA-MultiSumm algorithm is as follows: population size ps = 70, optimization probability op = 0.25, summary length maximum slm = 275 (during the evolutionary process), lambda λ = 0.84 and maximum number of optimizations $maxnumop$ = 20 (maximum number that an agent is optimized). A further parameter handled in the pre-processing stage is known as the sentence threshold, which ensures that each sentence of the summary has a minimum similarity to the document collection. The number of evaluations of the objective function was set at 15000. The algorithm was implemented on a PC Intel Core I3 2.99GHz CPU with 3GB of RAM in Windows 7.

Regarding the objective function, the process of tuning the weights of the MA-MultiSumm objective function was divided into two stages. In the first, a genetic algorithm (GA) was designed in order to obtain various weight ranges with which the objective function was then evaluated with the MA-MultiSumm algorithm to determine the best combinations of weights. In the second stage, the best set of weights obtained in the first stage is used as a reference to generate new sets of weights that are evaluated in order to obtain a better performance of the objective function.

4.2 Results

Table 1 presents the results obtained in ROUGE-1, ROUGE-2 and ROUGE-SU4 measures, for MA-MultiSumm and other state-of-the-art methods for the DUC2005 and DUCC2006 data sets. The best solution is represented in bold type. The number in parenthesis in the table shows the ranking of each method. As shown in this table, MA-MultiSumm improves upon the others methods in all ROUGE measures for DUC2005. MA-MultiSumm improves performance of DESAMC+DocSum by 1.63% for ROUGE-1, 5.72% for ROUGE-2 and 1.13% for ROUGE-SU4.

Table 1. ROUGE values of the methods on DUC2005 and DUC2006

	DUC2005						DUC2006					
Method	ROUGE-1		ROUGE-2		ROUGE-SU4		ROUGE-1		ROUGE-2		ROUGE-SU4	
DESAMC+DocSum	0.3937	(2)	0.0822	(2)	0.1418	(2)	**0.4345**	**(1)**	**0.0989**	**(1)**	**0.1569**	**(1)**
MA-MultiSumm	**0.4001**	**(1)**	**0.0868**	**(1)**	**0.1434**	**(1)**	0.4195	(5)	0.0986	(2)	0.1526	(4)
PLSA	0.3913	(3)	0.0811	(3)	0.1389	(5)	0.4328	(2)	0.0970	(3)	0.1557	(2)
LFIPP	0.3905	(4)	0.0804	(4)	0.1403	(3)	0.4209	(4)	0.0934	(4)	0.1534	(3)
MCMR	0.3891	(5)	0.0790	(6)	0.1392	(4)	0.4184	(6)	0.0928	(5)	0.1512	(5)
HybHSum	0.3812	(8)	0.0749	(8)	0.1354	(7)	0.4300	(3)	0.0910	(10)	0.1510	(6)
LEX	0.3760	(10)	0.0735	(10)	0.1316	(10)	0.4030	(9)	0.0913	(8)	0.1449	(10)
SVR	0.3849	(7)	0.0757	(7)	0.1335	(8)	0.4018	(10)	0.0926	(6)	0.1485	(8)
iRANK	0.3880	(6)	0.0802	(5)	0.1373	(6)	0.4032	(8)	0.0912	(9)	0.1450	(9)
HierSum	0.3753	(11)	0.0745	(9)	0.1324	(9)	0.4010	(11)	0.0860	(11)	0.1430	(11)
Centroid	0.3535	(12)	0.0638	(12)	0.1198	(12)	0.3807	(13)	0.0785	(13)	0.1330	(13)
SNMF +SLSS	0.3501	(13)	0.0604	(13)	0.1172	(13)	0.3955	(12)	0.0855	(12)	0.1429	(12)
TMR	0.3775	(9)	0.0715	(11)	0.1304	(11)	0.4063	(7)	0.0913	(7)	0.1504	(7)
MMR	0.3479	(14)	0.0601	(14)	0.1134	(14)	0.3716	(14)	0.0757	(14)	0.1308	(14)

With the DUC2006 dataset, the results of the evaluation show that the DESAMC+DocSum method is the only one that outperforms the proposed MA-MultiSumm algorithm in the ROUGE-2 measure. In the ROUGE-1 measure, MA-MultiSumm is outperformed by DESAMC+DocSum, PLSA, HybHSum and LFIPP. In the case of ROUGE-SU4, it is outperformed by the DESAMC+DocSum, PLSA and LFIPP methods. In summary, DESAMC+DocSum exceeds MA-MultiSumm by 3.67% for ROUGE-1. For ROUGE-2, the difference between these two methods is 0.30%, and for ROUGE-SU4 it better by 2.82%.

Because the results do not identify which method gets the best results on both data sets, a unified ranking of all methods is presented, taking into account the position each method occupies for each measure. Table 2 shows the unified ranking. The resultant rank in this table (last column) was computed according to the formula in Eq. (8).

$$Ran(method) = \sum_{r=1}^{14} \frac{(14 - r + 1)R_r}{14} \tag{8}$$

Where R_r denotes the number of times the method appears in the r-th rank. The denominator 14 corresponds to the number of methods with which the comparison was made.

Table 2. The resultant rank of the methods

Methods	R_r = 1	2	3	4	5	6	7	8	9	10	11	12	13	14	Rank
DESAMC+DocSum	3	3	0	0	0	0	0	0	0	0	0	0	0	0	5.8
MA-MultiSumm	3	1	0	1	1	0	0	0	0	0	0	0	0	0	5.4
PLSA	0	2	3	0	1	0	0	0	0	0	0	0	0	0	5.1
LFIPP	0	0	2	4	0	0	0	0	0	0	0	0	0	0	4.9
MCMR	0	0	0	1	3	2	0	0	0	0	0	0	0	0	2.9
HybHSum	0	0	1	0	0	1	1	2	0	1	0	0	0	0	2.6
LEX	0	0	0	0	0	0	0	1	1	4	0	0	0	0	2.4
SVR	0	0	0	0	0	1	2	2	0	1	0	0	0	0	2.1
iRANK	0	0	0	0	1	2	0	1	2	0	0	0	0	0	2.1
HierSum	0	0	0	0	0	0	0	0	2	0	4	0	0	0	2.0
Centroid	0	0	0	0	0	0	0	0	0	0	0	3	3	0	1.1
SNMF +SLSS	0	0	0	0	0	0	0	0	0	0	0	3	3	0	1.1
TMR	0	0	0	0	0	0	3	0	1	0	2	0	0	0	1.0
MMR	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0.4

Considering the results of Table 2, the following can be observed:

- The DESAMC+DocSum method takes first place in the ranking, focusing optimization on a sentence clustering problem. During the evolutionary process it carries out 50000 evaluations of the objective function.
- The MA-MultiSumm method takes second place in the ranking, but, DESAMC+DocSum used more than thirty times evaluations of the objective function than MA-MultiSumm. MA-MultiSumm outperforms methods based on clustering and probabilistic models such as PLSA (third place in the ranking) - a probabilistic model that applies the clustering technique - and HybHSum (sixth) that uses a probabilistic model to obtain the topics and then machine learning to train with a linear regression model; it outperforms evolutionary models such as LFIPP (fourth) that is based on a differential evolution model that represents the problem with the sentences of the summary and carries out 50000 evaluations of the objective function; and it outperforms MCMR (fifth) that is based on the binary particle swarm optimization model that also carries out 15000 evaluations of the objective

function, but this function is more expensive because uses the Google and cosine similarity measures.

- LEX is a method that uses clustering of terms and outperforms some probabilistic, algebraic reduction, and ranking-based methods.
- The SVR and iRank methods occupy an identical position in the ranking despite the fact that SVR is a method of algebraic reduction and iRank combines two methods of ranking that provide feedback for each other.
- The Centroid and SNMF+SLSS methods are placed equal in the rankings with a very similar performance in both data sets, despite the fact that Centroid carries out centroid-based clustering. SNMF+SLSS carries out sentence level semantic analysis (SLSS) and then symmetric non-negative matrix factorization (SNMF).
- TMR outperforms only MMR, although uses a probabilistic model for estimating the distribution of the topics and then machine learning for multinomial estimation, similar to HybHSum that ranks sixth.
- MMR comes last in the rankings, obtaining the worst results for the two sets of data in all the ROUGE measures used.

The experimental results indicate that optimization that combines global search based on population (CHC) with a heuristic local search for some of the agents (greedy search) - as is the case with the MA-MultiSumm memetic algorithm - is a promising area of research for the problem of generating summaries for multiple documents. This is because although the proposed algorithm takes second place in the ranking, the method that outperforms it (DESAMC+DocSum) involves 50000 evaluations of the objective function, exceeding at 30 times the evaluations of MA-MultiSumm (50000 vs 1500). So, given that the objective functions used are quite similar for the two methods, this implies a longer running time for the algorithm when compared with the MA-MultiSumm method.

In the proposed method, representation of the solutions is binary, indicating the presence or absence of the sentence in the summary, while in the case of the DESAMC+DocSum method, representation is real, indicating the group to which the sentence belongs. A process is then undertaken for the selection of sentences that make up the summary. This requires the DESAMC+DocSum method to carry out an additional process to obtain the summary, a process not required in the case of MA-MultiSumm.

5 Conclusions and Future Work

This paper proposes a memetic algorithm for automatically generating extractives summaries of multiple documents (MA-MultiSumm) based on CHC and greedy search, which prevents incest in calculating the hamming distance between the agent father and the agent mother. This it does by means of a threshold, whose value is smaller than that of the original CHC algorithm. It is noted that in this problem the agent is represented using many zeros and few ones, therefore the agents are similar to one another. The cross is made by means of the HUX scheme and optimization of the agents generated is done using local search. When cataclysm occurs, the two best individuals are preserved and the remaining individuals are created randomly.

The MA-MultiSumm method proposed was evaluated by means of ROUGE-1, ROUGE-2, and ROUGE-SU4 measures. When compared against other state of the art evolutionary methods on the data set DUC2005, the MA-MultiSumm method surpasses all methods in all measures. The DESAMC+DocSum method that takes second place is outperformed by 1.63% with ROUGE-1, 5.72% with ROUGE-2, and 1.13% with ROUGE-SU4. As regards the DUC2006 dataset, DESAMC+DocSum exceeds all methods in all measures. MA-MultiSumm with ROUGE-2 is outperformed by 0.30%; with ROUGE-1 is outperformed by 3.67%; and ROUGE-SU4 is outperformed by 2.82%.

In the unified ranking performed with all methods, the MA-MultiSumm method ranks second, behind only DESAMC+DocSum. However, this result is promising, given that the difference is minimal and, since the first makes 1500 evaluations of the objective function while the latter carries out 50000, runtime of MA-MultiSumm is shorter than for DESAMC+DocSum. In addition, while the latter represents a clustering problem and a subsequent process must be gone through to choose the sentences for the summary, in the case of MA-MultiSumm the sentences of the summary are taken directly from the best solution obtained following the running of the memetic algorithm. The MA-MultiSumm algorithm performed better in all measures with respect to the different methods used in automatic text summarization, such as graph-based, algebraic reduction, probabilistic, machine learning, and centroid.

Regarding results obtained in the task of automatically generating summaries using memetic algorithms, the use of these in this type of problem is promising, but it is necessary to continue to conduct research in order to achieve better results than those obtained in this article. Considering possible future work, it is necessary to carry out experiments on other data sets, to include other characteristics in the objective function that allow sentences relevant to the content of the documents and a summary that is closer to the reference summaries to be obtained and taking account other similarity measures like soft cosine measure [35]. Furthermore local search algorithms should also be explored, taking into account the specific characteristics of the automatic generation of summaries and thus enabling better results to be obtained.

Acknowledgments. The work in this paper was supported by a Research Grant from the Vicerrectoría de Investigaciones of the Universidad del Cauca and the Universidad Nacional de Colombia. We are especially grateful to Colin McLachlan for suggestions relating to English text.

References

1. Lloret, E., Palomar, M.: Text summarisation in progress: a literature review. Artificial Intelligence Review 37(1), 1–41 (2012)
2. Nenkova, A., McKeown, K.: A Survey of Text Summarization Techniques. In: Aggarwal, C.C., Zhai, C. (eds.) Mining Text Data, pp. 43–76. Springer, US (2012)

3. Miranda, S., Gelbukh, A., Sidorov, G.: Generación de resúmenes por medio de síntesis de grafos conceptuales. Revista Signos. Estudios de Lingüística 47(86) (2014)
4. Amini, M.-R., Usunier, N.: Incorporating prior knowledge into a transductive ranking algorithm for multi-document summarization. In: Proceedings of 32nd Annual ACM SIGIR Conference on Research and Development in Information Retrieval, Boston, USA, pp. 704–705. ACM (2009)
5. Ouyang, Y., et al.: Applying regression models to query-focused multi-document summarization. Information Processing & Management 47(2), 227–237 (2011)
6. Chen, Y.-M., Wang, X.-L., Liu, B.-Q.: Multi-document summarization based on lexical chains. In: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, Guangzhou, China, pp. 1937–1942. IEEE (2005)
7. Atkinson, J., Munoz, R.: Rhetorics-based multi-document summarization. Expert Systems with Applications 40(11), 4346–4352 (2013)
8. Otterbacher, J., Erkan, G., Radev, D.R.: Biased LexRank: passage retrieval using random walks with question-based priors. Information Processing and Management 45(1), 42–54 (2009)
9. Wei, F., et al.: 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, pp. 283–290. ACM (2008)
10. Radev, D.R., et al.: Centroid-based summarization of multiple documents. Information Processing & Management 40(6), 919–938 (2004)
11. Steinberger, J., Křišťan, M.: LSA-Based Multi-Document Summarization. In: Proceedings of 8th International PhD Workshop on Systems and Control, Balatonfured, Hungary (2007)
12. Sun, P., ByungRae, C.: Query-Based Multi-Document Summarization Using Non-Negative Semantic Feature and NMF Clustering. In: Proceedings Fourth International Conference on Networked Computing and Advanced Information Management, NCM, Gyeongju, pp. 609–614. IEEE (2008)
13. Hennig, L.: Topic-based Multi-Document Summarization with Probabilistic Latent Semantic Analysis. In: Proceedings International Conference RANLP, Borovets, Bulgaria, pp. 144–149 (2009)
14. Mei, J.-P., Chen, L.: SumCR: a new subtopic-based extractive approach for text summarization. Knowledge and Information Systems 31(3), 527–545 (2012)
15. Alguliev, R.M., et al.: MCMR: Maximum coverage and minimum redundant text summarization model. Expert Systems with Applications 38, 14514–14522 (2011)
16. Alguliev, R.M., Aliguliyev, R.M., Isazade, N.R.: DESAMC+DocSum: Differential evolution with self-adaptive mutation and crossover parameters for multi-document summarization. Knowledge-Based Systems 36(0), 21–38 (2012)
17. Abuobieda, A., Salim, N., Kumar, Y.J., Osman, A.H.: An Improved Evolutionary Algorithm for Extractive Text Summarization. In: Selamat, A., Nguyen, N.T., Haron, H., et al. (eds.) ACIIDS 2013, Part II. LNCS, vol. 7803, pp. 78–89. Springer, Heidelberg (2013)
18. Mendoza, M., et al.: Extractive single-document summarization based on genetic operators and guided local search. Expert Systems with Applications 41(9), 4158–4169 (2014)
19. Neri, F., Cotta, C.: Memetic algorithms and memetic computing optimization: A literature review. Swarm and Evolutionary Computation 2(0), 1–14 (2012)
20. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)

21. Hachey, B., Murray, G., Reitter, D.: The Embra System at DUC 2005: Query-oriented Multi-document Summarization with a Very Large Latent Semantic Space. In: Proceedings of the Document Understanding Conference (DUC), Vancouver, Canada (2005)
22. Silla, C.N., Pappa, G.L., Freitas, A.A., Kaestner, C.A.A.: Automatic text summarization with genetic algorithm-based attribute selection. In: Lemaître, C., Reyes, C.A., González, J.A. (eds.) IBERAMIA 2004. LNCS (LNAI), vol. 3315, pp. 305–314. Springer, Heidelberg (2004)
23. Ochoa, G., Verel, S., Tomassini, M.: First-improvement vs. Best-improvement local optima networks of NK landscapes. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN XI. LNCS, vol. 6238, pp. 104–113. Springer, Heidelberg (2010)
24. Lin, C.-Y.: Rouge: a package for automatic evaluation of summaries. In: Proceedings of the ACL-04 Workshop on Text Summarization Branches Out, Barcelona, Spain (2004)
25. Alguliev, R.M., Aliguliyev, R.M., Mehdiyev, C.A.: Sentence selection for generic document summarization using an adaptive differential evolution algorithm. Swarm and Evolutionary Computation 1(4), 213–222 (2011)
26. Celikyilmaz, A., Hakkani-Tur, D.: A Hybrid Hierarchical Model for Multi-Document Summarization. In: Proceedings 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, pp. 815–824. Association for Computational Linguistics (2010)
27. Lei, H., et al.: Modeling Document Summarization as Multi-objective Optimization. In: Third International Symposium on Intelligent Information Technology and Security Informatics (IITSI), China, pp. 382–386. IEEE (2010)
28. Wei, F., Li, W., Liu, S.: iRANK: a rank-learn-combine framework for unsupervised ensemble ranking. American Society for Information Science and Technology 61(6), 1232–1243 (2010)
29. Haghighi, A., Vanderwende, L.: Exploring content models for multi-document summarization. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Boulder, Colorado, pp. 362–370. Association for Computational Linguistics (2009)
30. Wang, D., et al.: 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, pp. 307–314 (2008)
31. Tang, J., Yao, L., Chen, D.: Multi-topic based query-oriented summarization. In: Proceedings of the Ninth SIAM International Conference on Data Mining, Nevada, USA, pp. 1148–1159 (2009)
32. Carbonell, J., Goldstein, J.: The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and development in Information Retrieval, Melbourne, Australia, pp. 335–336. ACM (1998)
33. Eiben, A.E., Smit, S.K.: Evolutionary Algorithm Parameters and Methods to Tune Them. In: Hamadi, Y., Monfroy, E., Saubion, F. (eds.) Autonomous Search, pp. 15–36. Springer, Heidelberg (2012)
34. Cobos, C., Estupiñán, D., Pérez, J.: GHS + LEM: Global-best Harmony Search using learnable evolution models. Applied Mathematics and Computation 218(6), 2558–2578 (2011)
35. Sidorov, G., et al.: Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model. Computación y Sistemas 18(3) (2014)