Optimization of Boolean Queries in Information Retrieval Systems Using Genetic Algorithms - Genetic Programming and Fuzzy Logic

Suhail S. J. Owais
Ajloun University College - 'AL-Balaqa' University
suhailowais@yahoo.com

Journal of Digital Information Management

***ABSTRACT**: This paper proposes to use two information retrieval system models (Boolean information retrieval model and extended Boolean (fuzzy) information retrieval model). These models differ by using Boolean queries or fuzzy weighted queries. It also proposes a way for optimizing user query for the two models by using genetic programming and fuzzy logic. And proposes to use more number of Boolean operators (AND, OR, XOR, OF, and NOT) instead of the standard Boolean operators (AND, OR, and NOT), and use weights for Boolean operators and for terms in fuzzy models.*

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval]; **J.3** [Life and Medical Scieces]: Biology and genetics

General Terms
Boolean operators, Fuzzy model, Information retrieval

Keywords: Boolean Query, Information Retrieval, Genetic Algorithms, Genetic Programming, Fuzzy Logic, Term Weights, and Boolean Operator Weights.

Received 8 April 2006; Reviewed and accepted 24 April 2006

1. Introduction

One of the most pressing issues with today's explosive growth of the Internet is so-called resource discovery problem [3]. That is how to find information interest among the vast and growing amount of information available. One of the most important uses of the public network is to find suitable information for such user query request. In this paper, we discuss the use of weights for both Boolean operators and the use of more number of Boolean operators to optimize the user query. We work on two IR models (Boolean and fuzzy) to optimize the user query using one of the evolutionary algorithms – genetic programming and fuzzy logic.

Genetic algorithm was implemented in both models, but fuzzy logic was used only in fuzzy or extended Boolean model. For the both models harmonic mean measure was used to measure the IR performance. Harmonic mean was used to combine precision and recall measures both at once to improve the IR performance.

2. Motivation

Because of the widespread use of web search techniques, particularly in academics, search processes need to be understood. Many users of web are not well trained in Boolean algebra; *"the problem of learning the correct interpretations of Boolean operators and their rules of precedence"* [1]. Hence, the motivation of this current work is to produce two IRs models, which enable to optimize the user query. The optimized query will retrieve the most relevant documents with less number of non-relevant documents to his/her search query. The deployment of Boolean operators (*AND, OR, XOR, OF,* and *NOT*) using harmonic mean measure improves the performance of IR.

3. Related work

The body of literature in information retrieval is filled with many papers. Generic algorithms offer more promise than rest. Masaharu et al. [17], propose to use an IR interface that employed a few number of query terms and concept categories with Boolean expressions; they use only the words that exist in the original query for reformulating the Boolean query; and their work is confined to two Boolean operators only. Cordon et al. [18], represent the query in a parse tree with maximum of 20 nodes; where they used only "AND, OR, and NOT" Boolean operators, and moreover the testing is limited to a small set of 400 documents only. The study of Kraft et al. [16] has addressed the genetic programming where they optimize user search queries and investigate whether precision or recall is more efficient objective function and presented experiments over non-fuzzy collection of documents. Cordon et al. [19] propose the use of multi objective evolutionary algorithms (EA), and they offer comparison of several EA oriented approaches for optimization of persistent search queries. Among the aforesaid studies the works of Kraft and Cordon offer more promises.

4. Information Retrieval

IR is the process of extracting useful information from databases of text documents (collection of document) via word or term searches and other techniques. Search queries extract and produce the results that meet user criteria.

4.1 IR Concepts

Documents and queries are two major types of data in an IRS. IR deals with representation, storage, organization of, and access to information item [2]. IRs deals with databases that include information items and documents and consist textual, pictorial or vocal information.

Many IR activities are based on Boolean queries. Query terms are basically combined by the logical operators such as AND, OR. A query is an expression of information need. A query is submitted to the system, which aims to find information relevant to the information need expressed in the query [9].

The goal of IRs is to lead "the user to those documents that will best enable him/her to satisfy his/her need for information" [4]. The similarity between a query and documents is measured by different retrieval strategies. These measures are based on the more frequent terms found in both documents and a query. The more relevant documents are deemed to be the user query request.

4.2 Structural and Semistructural Data

In the recent years, the amount of heterogeneous data available electronically has increased dramatically. The data resides in different forms, unstructured data in file systems, and highly structured in relational database systems.

4.2.1 Structured Data

Traditionally, IR techniques focused on retrieving information from unstructured documents. The focus is now shifting to retrieving information from structured documents.

Structured retrieval deals with the retrieval of documents for a query with structure. Structure is added by using any structured language to represent queries. Hence, we need structured queries and documents to perform retrieval tasks and conduct experiments. Effective retrieval of structured documents should exploit the content and structural knowledge associated with the documents. This knowledge can be used to focus retrieval to the best entry point: document components that contain relevant information [5].

4.2.2 Semistructured Data

Semistructured data is data that presents some regularity but perhaps not as much as some relational data. The data is managed under an increasing diversity of data models and access mechanisms. Data is accessible through a variety of interfaces including Web browsers; databases query languages; application-specific interfaces; or data exchange format [7]. Much of these data are semistructured ones. Query languages are inadequate for querying semistructured data. Such examples in which semistructured data arise are when data is stored in sources that do not impose a rigid structure (such as World Wide Web) [6, 20].

4.3 Information Retrieval Performance

Evaluation of an IRS, is measured basically by effectiveness for which two statistics, precision and recall are used . These measures are evaluated over a collection of documents. The collection was divided into subsets of documents (*Relevant set, Retrieved set, Relevant-Retrieved set, and the not Relevant-Retrieved set*) with respect to the user query as shown in figure 1.

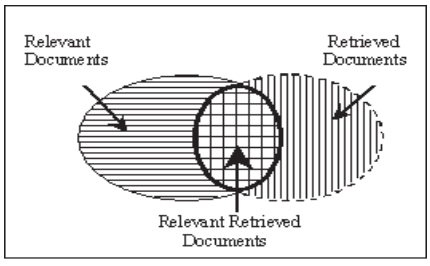

Figure 1: Collection of Documents

IR evaluation measures play an important part in the development of retrieval systems as it stimulates their improvement. Precision measure is the percentages of the retrieved documents that are relevant to the user query. It pertains to a system's ability to retrieve as many relevant documents as possible. Recall measure is the percentage of the relevant documents that are retrieved for the given query. It pertains to a system's ability to retrieve as few non-relevant documents as possible.

The following definitions are used in computing precision and recall measures:-

Retr : the set of retrieved documents.
Rel : the set of relevant documents.

Note that *Retr* and *Rel* could be disjunction or contain exactly the same set of documents. Recall and precision measures were defined as follows:

$$\textbf{Precision} = \frac{|Rel \cap Retr|}{|Retr|} \quad ; \quad \textbf{Recall} = \frac{|Rel \cap Retr|}{|Rel|}$$

There is an tradeoff relationship between precision and recall. Retrieving most of relevant documents increases both precision and recall. Retrieval of nonrelevant documents decreases precision but does not affect recall.

By combining both measures precision and recall will produce better performance by using the harmonic mean measure "$F - score$". It combines recall and precision measures into a single one. Measure ensures that it will have values within an interval [0, 1] [26] and it is defined as follows:

$$F - score = \frac{2 \times Precsion \times Recall}{Precsion \times Recall}$$

5. Evolutionary Algorithms

EA is a metaheuristic optimization algorithm which incorporates aspects of natural selection or survival of the fittest that uses mechanisms from biological evolution. Such mechanisms are selection, crossover, mutation, recombination and survival (*genetic operators*). EA maintains a population of solutions to problem which are usually generated randomly and initialized to be an initial population. The population evolves according to rules of EA are a general adaptable concept for solving difficult problems with better results in optimization area.

EA approaches are genetic algorithm; genetic programming; evolutionary strategy; and evolutionary programming.

5.1 Genetic Algorithms

Genetic algorithm (GA) is a refined and improved search technique used in information retrieval. GA implements as a computer simulation to find approximate solutions to combinatorial optimization problems. The most common application of GA is function optimizer.

GA is based on natural selection, the *survival-of-the fittest*. GA evolves a population of chromosomes or individuals as the process of natural selection. And during its process, it generates an offspring (new chromosome/s) using genetic operators "*selection, crossover and mutation*", where objective function used for evaluate each chromosome in the population.

GA consists of population of chromosomes that reproduced over generations according to their fitness in an environment. The chromosomes that are most fit are most likely to survive, mate, and bear children (offspring) [8]. Figure 2 shows the basic structure of GA.

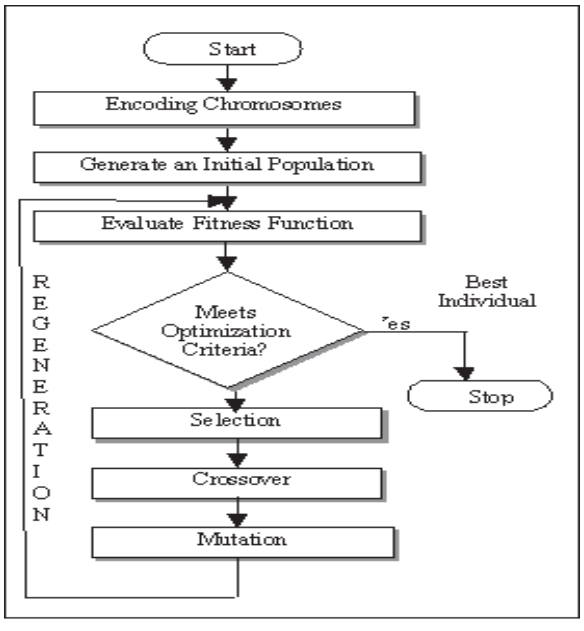

Figure 2: Genetic Algorithm Process

The GA process can be terminated by stopping its execution which determines its terminated conditions. These conditions may be specified as a maximum number of generations; or it attains an acceptable fitness level; or it happens when there are no improvements over some known generations.

6. Boolean Query and Boolean Operators

The user perceives a query as a list of terms irrespective of the language used to construct. The Boolean query is based on concepts from logic, or Boolean algebra. A query consist of set of terms joined together by logical connectives like AND, OR, and NOT Boolean operators.

Another set of Boolean operators especially XOR, and OF operators are proposed in this paper. These two operators are not used in most of Boolean queries.

XOR Boolean operator presented here uses the standard Boolean operators OR, AND, and NOT as follows:

$term_1$ XOR $term_2 \equiv$ (NOT $term_1$ AND $term_2$) OR ($term_1$ AND NOT $term_2$)

OF Boolean operator allows the user to specify how many terms from a list of terms must be present if it is not necessary to have all terms in the list. For example, a user may specify four terms in a query but he needs to have any two of them. It demonstrates using OF Boolean operator and in standard Boolean operators.

2 OF ($term_1$, $term_2$, $term_3$, $term_4$) $\equiv$
($term_1$ AND $term_2$) OR ($term_1$ AND $term_3$) OR ($term_1$ AND $term_4$) OR ($term_2$ AND $term_3$) OR ($term_2$ AND $term_4$) OR ($term_3$ AND $term_4$)

6.1 Term and Boolean Operator Weights

The importance for such a term is different than others in a document or in a Boolean query. Not all terms have the same equality of importance in the documents or in the queries. And at the same time, Boolean operators have varying significance in the query. Thus, it is important to allow the user to indicate the relative importance of various terms and Boolean operators by weighing them.

Weighting of terms modifies the calculations upon which relevance judgments are made [1]. While, assigning weights to document terms is a complex process [1], it should be within the range of values; such as [0, 1].

Also there are weighs for Boolean operators in queries, where each Boolean operator has a weight within range of values as in terms weights; such as [0, 1].

7. Fuzzy Logic

Fuzzy logic was developed by L. A. Zadeh in the mid-1960s for representing uncertain and imprecise Knowledge [10, 11, 12, 13, 15, 23]. Fuzzy Logic is a form of mathematical technique employing a range of values from "*true*" to "*false*" for problems that have many solutions rather than one.

Fuzzy logic works with ranges of values, and dealing with the concept of *partial truth*. Fuzzy logic replaces Boolean truth values {*true, false*} or {0, 1} with degrees of truth [0,1]. Fuzzy truth represents membership in vaguely defined sets. Set membership values between 0 and 1 and including them [0, 1].

7.1. Fuzzy Set and Operators

Zadeh introduces the concept of a fuzzy set whose boundary is not sharp or precise [10, 22]. He defined the probability of a fuzzy event by a Lebesgue integral [14].

A classical set A in X is a set of ordered pairs which it defined by indicator function $I_A(x)$, and defined as follows:

$$A = \{(x, I_A(x)) \mid x \in X\}, \quad I_A(x) \in \{0,1\}$$

A fuzzy set A in X is a set of ordered pairs which it defined by membership function $\mu_A(x)$, and defined as follows:

$$A = \{(x, \mu_A(x)) \mid x \in X\}, \quad \mu_A(x) \in [0,1]$$

Measure the fuzzy events by the defined by fuzzy t-conorm (triangular conorm) which it is integral of the fuzzy membership function. Each element has an associated *membership degree* with respect to a given set, which represent in some sense of strength or degree of belief in its membership in the set [1].

For The two fuzzy sets A and B defined the two membership functions $m_A(x)$ and $m_B(x)$. The standard fuzzy operators are defined as follows:

- *union* → $\mu_{(A \vee B)}(x) = \max\{\mu_A(x), \mu_B(x)\}$
- *intersection* → $\mu_{(A \wedge B)}(x) = \min\{\mu_A(x), \mu_B(x)\}$
- *complement* → $\mu_{\neg A}(x) = 1 - \mu_A(x)$

Function that qualifies as fuzzy unions and fuzzy intersections are usually referred to the literatures as "*t-conorm*" and "*t-norm*" respectively.

7.2. Fuzzy Aggregation Operator

For multi-criterion decision and ranking the AND and OR operators can interpret as a fuzzy-min (fuzzy intersections) and fuzzy-max (fuzzy unions) functions respectively [28].

Fuzzy querying and ranking is a flexible process through which linguistic concepts can be used in the queries and ranking in a very natural form [22]. Thus, fuzzy query is one solution to the problems of presenting a query that accurately represents information needs. Aggregating operations defined as function of n arguments; where $n > 2$. So it is an operation by which several fuzzy sets are combined in desirable way to produce single fuzzy set.

8. Evolutionary Learning for Optimizing Queries in IRs

IRs works on filtering of collection of documents for answering user query or Boolean query. It finds the most relevant documents to be retrieved with minimum number of non-relevant documents. There are many problems with Boolean retrieval systems [1].

Boolean operators and their rules of precedence underlying are not properly conceived and their potential is not realized completely in IR. Our work uses the GP to optimize the Boolean query for the IR models, and allows the user to indicate the relative importance of various terms by weighing them and weighing the Boolean operators. And we use fuzzy theorem in our work.

8.1 IRs Structure

The two IR models were developed by using GP techniques to optimize the user query. The systems were tested by implementing several experiments for Boolean and fuzzy IR models.

In the system, D is the collection of documents, Q is the set of queries, and T is the set of all distinct terms in D, where

$$D = \{D_1, D_2, \ldots, D_N\},$$
$$Q = \{Q_1, Q_2, \ldots, Q_M\},$$
$$T = \{T_1, T_2, \ldots, T_W\}$$

where N, M, and W represents the number of documents in the collection, the number of the queries, and the number of distinct terms respectively.

Each document was described by a set of distinct terms. The description for the j^{th} document D_j in the collection is as follows:

$$D_j - \{t_1, t_2, \ldots t_{N_j}\}, \quad t_i \in T$$

where, N_j is the number of distinct terms in document D_j.

8.2. Boolean Indexing

Boolean IR model provides a crisp representation of information content of the document [25]. The value for each term will be 1 if this term exists in the document or 0 if not.

The Boolean indexing function (*membership function*) F maps a given index term T and document D, the main formula for F is defined as follows:

$$F : D \times T \rightarrow \{0,1\}$$

8.3 Fuzzy Indexing and Fuzzy Queries

Boolean retrieval extended by fuzzy set theory is used to improve the Boolean retrieval model. The fuzzy indexing function F allowed taking values from the interval [0, 1] by implementing the fuzzy set theory.

Fuzzy indexing represents the term weight in such document. Fuzzy indexing function $F(D_j, T_i)$ represents the weight for the term T_i in the document D_j. The term significance $0 \leq F(D_j, T_i) \leq 1,0$ implies no significance and 1 means total significance and any other value is partially significant. The main formula used for the fuzzy indexing function F is as follows:

$$F : D \times T \rightarrow [0,1]$$

The fuzzy representation for a document represented as a fuzzy set of terms is as follows:

$$R(D_j) = \{Ti, F(D_j, T_i)) \mid T_i \in T, D_j \in D\}$$

The definition of the criteria for an automatic computation $F(D_j, T_i)$ is a crucial aspect and generally this value is defined on the basis of statistical measurements with the aim of optimizing retrieval performance [25]. Several indexing functions were proposed for estimating the fuzzy term weights in documents. In our work, fuzzy indexing function was implemented according to Kraft et al. [25].

The general definition for the indexing function F based on normalized frequency of term in a document and on normalized inverted document frequency (IDF) which is defined as follows:

$$F(D_j, T_i) = f_{D_j T_i} \cdot IDF_{T_i}$$

where $f_{D_j T_i}$ is a normalized term frequency of term T_i in the document D_j, and is a normalized IDF for the term T_i. There is another normalized function g which is used for estimating the value of . The general formals for $f_{D_j T_i}$ and IDF_{T_i} are as follows.

$$f_{D_j T_i} = \frac{freq(T_i, D_j)}{MAX(freq(T_l, D_j))}$$

$$IDF_{Ti} = g(\log(N/Nt))$$

Where, N_t is the number of documents that contains at least one occurrence of term t.

Term weight increases with multiple occurrences of the term in the document, and decreases with occurrence of the term in more documents [25].

Query term weight function *a* is a mapping function that's maps a given index term T_i in a given query Q_k, and it is defined as follows:

$$a: T \times Q \rightarrow [0, 1]$$

The importance for the search term T_i in the query Q_k will be estimated by a function $a(T_i, Q_k)$. Term weight T_i can be interpreted as threshold, or as perfect significance degree [25].

Evaluating the single weighted query term T_i with weight function *a*, with against of the index term weight $F(D_j, T_i)$ in the document D_j is defined by the function *g* as follows:

$$g : [0,1] \times [0,1] \rightarrow [0,1]$$

Where, evaluating function for an atomic selection condition in a query can be interpreted by the function g. The retrieval status value (RSV) for query Q_k consist of single selection criterion is defined as follows:

$$RSV = g(F(Dj, Ti), a(Ti, Qk))$$

Term with query term weight $a(T_i, Q_k)$ is meant as request for document satisfying $F(D_j, T_i) = a(T_i, Q_k)$, and the evaluation function was implemented with respect to the following relation:

$$g(F(D_j, T_i), a) = \begin{cases} P(a)\dfrac{F(Dj, Ti)}{a} & \text{when } F(Dj, Ti) < a \\ P(a) + Q(a)\dfrac{F(Dj, Ti) - a}{1 - a} & \text{Otherwise} \end{cases}$$

where *P(a)* and *Q(a)* are two factors used for adjusting *a* [17], and they are defined as follows:

$$P(a) = \frac{1 + a}{2}, \quad Q(a) = \frac{1 + a^2}{4}$$

8.4 GP Implements Genetic operators

Optimizing the user query in both Boolean and fuzzy IRs models were implemented using GP. GP starts processing on a pool of chromosomes or individuals called initial population and fixed in all generations. GP impellents the following genetic operators *Objective Function, Selection, Crossover and Mutation* operators.

8.4.1 Chromosome Encoding

Set of queries are presented as an initial population. They are mentioned as suggested solutions to the problem. These queries are in form of Boolean queries or in form of weighted queries with respect to IRs model.

Each chromosome from the initial population was encoded in form of tree structure data type. It was encoded from one of the suggested solutions, where terms as terminals and Boolean operators are non terminals. The weight for such term or for such Boolean operator appears directly after the term or the Boolean operator as ($term_1$:0.32) and for (NOT:0.6). *The Boolean operator or any term has no weight and thus its weight is* now 1.

For example on encoded chromosome for the following query was shown in figure 3:

($term_8$:0.3 AND $term_2$) OR (($term_8$ AND:0.7 $term_7$:0.8) OR ($term_2$ AND $term_7$))

8.4.2 Objective Functions

Objective (*evaluation or fitness*) function measure the performance of IRs with respect to the particular set of GA parameters. Fitness function evaluates chromosomes using *F -score* measure. The extended Boolean (fuzzy) IRs effectiveness' could not be measured in form of using for Boolean IRs model. The main measures precision and recall defined with help of the standard cardinality measures of fuzzy sets $\sum$–count [2]:

$$\rho(X|Y) = \begin{cases} \dfrac{\|X \cap Y\|}{\|Y\|} = \dfrac{\sum_i \min(x_i, y_i)}{\sum_i y_i}, & when \|Y\| \neq 0 \\ 1, & when \|Y\| = 0 \end{cases}$$

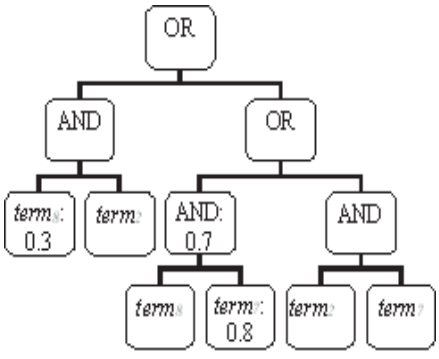

Figure 3: Encoded chromosome

where *X* and *Y* are two fuzzy sets.

For computing the new measures of precision and recall, *R* is the set of documents that are relevant to the query, and *A* is the set of documents that are considered as relevant by the system. They are defined as follows:

$precision = \rho(R|A)$ and $recall = \rho(A|R)$

And for computing, *F -score* measure was evaluated as defined before.

The indexing module realizes indexing for Boolean and fuzzy IR as described before. The core of IRs is based on Boolean and extended Boolean IR model.

8.4.3 Selection Operator

To produce an optimize query to the user query, several generations was done by GP process to get an optimal solution. GP starts within an initial population and this generation was called first generation. To pass the next generation, an intermediate generation will be passed through from the current generation to the next generation using selection operator.

The selection operator is based on the survival of the fitness by selecting best individual/s from the population. Two types of selections are used to select two chromosomes for recombination which are called as parent-1 and parent-2. These types are *elitism* and *probabilistic* selection.

After selection has been carried out, the construction of the intermediate generation was completed and ready for recombination to produce the next generation.

8.4.4 Crossover Operator

Single point crossover was used. From each parent one node randomly was selected with respect to crossover probability *pc* value. Then swap the sub-trees between the two parents to produce new offspring's, sub-tree was determined by the selected node and mentioned as the root for sub-tree.

The two new offspring's were generated and called offspring-1 and offspring-2.

8.4.5 Mutation Operator

Mutation operator processed over the generated offspring's to produce new chromosomes. Select one node randomly from each offspring with respect to mutation probability *pm* value; and mutate it using one of the following mutation methods:

- Mutate its weight *(for fuzzy case only)*
- Insert a new node with a unary operator before the selected node.
- If the selected node was a term node, mutate it into another term that chosen randomly from one of a specified list of terms from one of the following term lists:
  - collection terms list.
  - initial population terms list.
  - user query terms list.
  - from a user defined list of terms.

♦ If the selected node is a Boolean operator, mutate it into another Boolean operator from the same arity value and type. Arity value and type are explained in [24].

♦ If the selected node is a unary operator, eliminate it.

Figure 4 demonstrates an example on mutation offspring-1 by changing the weight for the Boolean operator (AND:0.7 ⟶ AND:0.9). And figure 5 demonstrates an example on mutation offspring-2 by changing a binary Boolean operator with other one (OR ⟶ AND).

8.4.6 Reinsertion

After producing offspring's they must be inserted into the current population by replacing them by the worst chromosomes. The new offspring's will be replaced by the worst chromosomes from the population in the current generation. Such inserting scheme was used by determining which offspring should be inserted into the new population and which chromosome from the population should be eliminate it. The new population will be used in the next generation.

8.4.7 Terminate Process

Several generations would be processed until reaching an optimized solution for the user query. Such mechanisms were defined for terminating GP execution. Such as, when reaching an optimal query to the user query (*F- score* = 1) or when executes number of

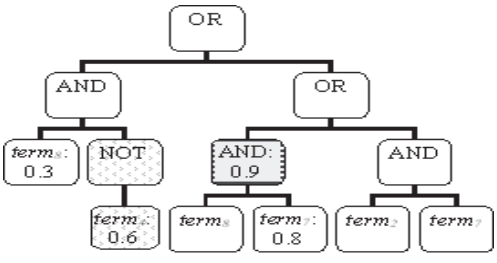

Figure 4: Mutate Offspring-1

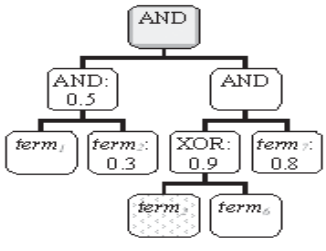

Figure 5: Mutate Offspring-2

generations that exceeds the maximum number of generations defined for termination, or when there are no improvements after number of generations.

8.5 System Environment

Several parameters should be defined for executiing the systems for testing. The most important parameters in GP that should be defined and fixed before executing the systems, are as follows:

- Population size, fixed in all generations⟶ 15
- Crossover probability value⟶ 0.8
- Mutation probability value⟶ 0.2
- Maximum number of generations⟶ 500
- Maximum number of generations without improvements⟶50

For the initial population, there are two cases are implemented for both Boolean IRs and fuzzy IRs models:

1. Case-1: all individuals for the initial population are generated randomly.
2. Case-2: some of the individuals are generated randomly and the rest of the initial population are added from previous solutions "*seeded*".

➢ For Boolean IRs model, the Seeded solutions to the initial population are:

"FUNDS" OR "BIBLIOGRAPHIC"
"EXTREMELY" AND "INNOVATORS"
NOT ("POOR" XOR "FUNDS")

➢ For fuzzy IRs model, the Seeded solutions to the initial population are:

"FUNDS":0.9 OR "BIBLIOGRAPHIC":0.8
"EXTREMELY":0.3 AND "INNOVATORS"
NOT ("POOR" XOR:0.03 "FUNDS":0.5)

- The user query used in testing both models are:
  - ➢ For Boolean IRs model:
    (("EXTEREMELY" AND "POOR") OR "FUNDS")
  - ➢ For fuzzy IRs model:
    (("EXTEREMELY":0.94 AND "POOR":0.5) OR:0.5 "FUNDS":0.9)

8.6 Experiments Tests and Results

The core of IRs is based on the Boolean and fuzzy IRs model. The systems were tested by implementing them several times. Each time an experiment was repeated 10 times and the average were taken for them for optimizing the user query.

The collection of documents that was used in our systems is LISA collection (*Library and Information Science Abstracts*). LISA is a public collection available at University of Glasgow. LISA has a size of 3.4 Mega bytes, and contains 5999 documents with 18442 unique indexed terms (distinct).

LISA collection was indexed for testing both models. In each experiment some documents from the collection were marked as relevant documents to the user query. Relevant documents depend on the user query that was implemented for the two cases Boolean IRs and fuzzy IRs models.

Eight cases tested the systems, with the following meanings- (*crisp*: Boolean IRs model, *extended*: fuzzy IRs model, *all*: all individuals for the initial population are randomly generated, *some*: some individuals from the initial population are randomly generated and the rest are seeded, and two types of selection mechanisms: *elitism* and *probabilistic*). The eight cases are described as follows:

1. *crisp, all,* and *elitism.*
2. *crisp, some,* and *elitism.*
3. *extended, all,* and *elitism.*
4. *extended, some,* and *elitism.*
5. *crisp, all,* and *probabilistic.*
6. *crisp, some,* and *probabilistic.*
7. *extended, all,* and *probabilistic.*
8. *extended, some,* and *probabilistic.*

The average results for the experiments were obtained from the systems tested and presented in figure 6. From the figure, we can find that the optimized query to the user query are obtained with high fitness value of F-score in all cases except two cases 3 and 7.

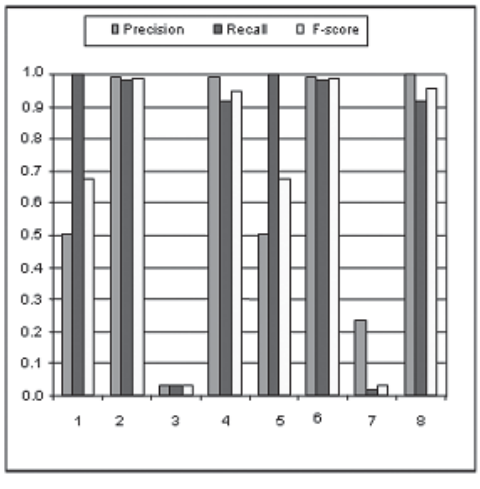

Figure 6: IRs models experiments results

9. Conclusions, and Future Work

From the experiments results, GP can be used as an effective optimization tool in IR areas to optimize the user query in Boolean IRs and in fuzzy IRs models.

The conclusions from testing the two IRs models are outlined as follows:

♦ For cases 1, 2, 5 and 6 for Boolean model, the produced optimized query were with high performance values for *F-score* which it is greater than 0.6. And with values more than 0.9 especially when the initial population was seeded with some optimal solutions.

♦ For cases 4 and 8 for fuzzy model and especially when the initial population was seeded with some optimal solutions, the optimal solutions were produced with high performance values of more than 0.9 than when no seeded done.

The fuzzy IRs did not reach the high values when it is implemented with initial population that had all individuals were randomly generated especially for the two cases 3 and 7. Hence, when some of the optimal solutions were seeded to the initial population the system produced a butter result which means that use optimal solutions in the initial population improves the system.

The values for precision, recall, and *F-score* reach the highest values especially when some of the optimal solutions are seeded to the initial population.

The highest value for the fitness function should be 1.0, when precision and recall reach the highest values 1.0. From the relationship curve for precision and recall, the ideal point value for both (*precision* and *recall*) is (1.0, 1.0). Use as fitness measure gives a priority to the highest possible values for both precision and recall both at once. So reaching the highest fitness values in most cases approximately, indicate that GP were successfully produced optimized queries to the user queries.

These results confirm the usability of GP in the task of search query optimization. The system will be improved when the optimized query will be based on user search queries and stored in the user profile file and used in the initial population.

Some comparisons' for our work with previous work are mentioned in the following points:

- More Boolean operators were used (*AND, OR, XOR, OF* and *NOT*) not like those used in [16, 17, 18, 19].
- A large collection of documents are used for testing IRs models in the current work instead of using smaller collections as in [16, 17, 18, 19, 21].
- Support of Boolean operators weights as an expression tool in fuzzy queries and experiments of queries without operator and term limit has value.

And for the future work, we plan to implement the variants of fuzzy concepts in IRs model such as use of different aggregation operators (i. e. OWA [17] and uninorms [27]). We also plan to implement more different types of measures to test the performance in IR like E measure, User-Oriented measures, etc.

References

[1] Korfhage, R. R (1997). *Information Storage and Retrieval*, John Wiley & Sons, Inc., USA.

[2] Baeza-Yates, R., Ribeiro-Neto. *B (1999). Modern Information Retrieval*, ACM press Addison Wesley, New York.

[3] Yuwono, B., Lee, D. L (1996). "WISE: A World Wide Web Resource Database System," *IEEE Transaction on Knowledge and Data Engineering* 8 (4) 548 - 554.

[4] Belkin, N. J., Croft, W B (1992). Information Filtering and Information Retrieval: Two Sides of the Same Coin? *Communications of the ACM* 35 (12) 29 - 38.

[5] Kazai, G., Lalmas, M., Rölleke, T (2001). A Model for the Representation and Focussed Retrieval of Structured Documents based in Fuzzy Aggregation," String Processing and Information retrieval (SPIRE) Conference, Laguna De San Rafael, Chile, p. 123 - 135.

[6] Quass, D., A. Rajaraman, A., Ullman, J., Wiom, J., Sagiv, Y (1997). "Querying Semistructured Heterogeneous Information," *Journal of Systems Integration* 7. 381 - 407.

[7] Abiteboul, S (2001). Semistructured Data: from Prictice to Theory", *In*: Proceedings of the 16th Annual IEEE Symposium on Logical in computer Society (LIC'01), p. 0379.

[8] Spears, W. M., Anand, V (1991). A Study of Crossover Operators in Genetic Programming," *In:* Proceedings of the Sixth International Symposium on Methodologies for Intelligent Systems. 409 - 418.

[9] Crestani, F., Lalmas, M., Van Rijsbergen, C.J., Campbell, I (1998). Is This Document Relevant?Probably: A Survey of Probabilistic Models in Information Retrieval," *ACM computing Surveys* 30 (4) 528 - 552.

[10] Klir, G.J (1997). *Fuzzy Set Theory: Foundation and Applications*, Prentice Hall PTR, USA.

[11] Jain, L.C., Martin, *N.M (1998). Fusion of Neural Networks, Fuzzy Systems and Genetic Algorithms: Industrial Applications*, CRC Press, 1998.

[12] Kuncheva, L. I.,Jain, J.W. L.C., Al-Zaidan, A (2001). A Fuzzy Model of Heavy Metal Loadings in Marine Environment", Da Ruan, J. Kacprzyk and M. Ferdrizzi (Eds) Soft computing for Risk Evaluation and Management, Springer-Verlag. p. 335 - 371.

[13] Kuncheva, L. I., Wrench, J., Jain, L.C., Al-Zaidan, A (2001). A Fuzzy Model of Heavy Metal Loadings in Marine Environment", Da Runa, J. Kacprzyk and M. Ferdrizzi (Eds), Soft Computing for Risk Evaluation and Management, Springer-Verlag. p. 355 - 371.

[14] Grabisch, M., Murofushi, T., Sugeno, M (1992). Fuzzy measures of fuzzy events defined by fuzzy integrals, *Fuzzy Sets and Systems* 50. 293 - 313.

[15] Zadeh, L. A (2002). From Computing With Numbers to Computing With Words – From Manipulation of Measurements to Manipulation of Perceptions, *International Journal of Applied Mathematics and Comput. Science* 12 (3) 307 - 324.

[16] Kraft, D. H., Petry, F. E., Buckles, B. P., Sadasivan, T (1997). "Genetic algorithms for query optimization in information retrieval: relevance feedback", *In*: E. Sanchez, T. Shibata, L.A. Zadeh, Genetic Algorithms and Fuzzy Logic Systems, World Scientific. 155 - 173.

[17] Masaharu, Y., Makoto, H (2005). An Appropriate Boolean Query Reformulation Interface for Information Retrieval Based on Adaptive Generalization,*In*: International Conference on Data Engineering. p. 148 - 153.

[18] Cordon, O., Herrera-Viedma, E., Luque, M (2002). "Evolutionary Learning of Boolean Queries by Multiobjective Genetic Programming." *In*: J.J. Merelo Guervos et al. (Eds.): PPSN VII, LNCS 2439, Springer-Verlag Berlin Heidelberg. p. 710 - 719

[19] Cordon, O., Moya, F., Zarco, C (2004). Fuzzy Logic and Multiobjective Evolutionary Algorithms as Soft Computing Tools for Persistent Query Learning in Text Retrieval Environment" *In*: IEEE International Conference on Fuzzy Systems, 2004, pp. 571 - 576.

[20] Abitebou, S., McHugh, J M., Vassalos, Rys V., Wiener, L.L (1998). Incremental Maintenance for Materialized Views over Semistructured Data" *In*: Proceedings of the International Conference on very Large Databases (VLDB), 1998, pp. 38 - 49.

[21] Owais, S. S.J., Kromer, P., Snasel, V (2005). Query Optimization by Genetic Algorithms, Database Texts Specifications Objects (DATESO) p. 125 - 137.

[22] Nikravesh, M., Azvine, B (2002). Fuzzy Queries, Search, and Decision Support System", *Soft Computing,* p. 373 -399.

[23] Zadeh, L. A (1976). A fuzzy-algorithmic approach to the definition of complex or imprecise concepts", *International Journal of Man-Machine Studies* 8. 249 - 291.

[24] Owais, S. S.J., Kromer, P., Snasel, V (2005). "Implementation of Evolutionary Algorithms-Genetic Programming for Optimization of Boolean Queries in Information Retrieval Systems", *GESTS International Transactions on Computer Science and Engineering*, 9 (1) 47 - 66.

[25] Kraft, D. H., Bordogna, G., Pasi, G (1999). *Fuzzy Set Techniques in Information Retrieval*, *In:* Bezdek, J.C., Didier, D. and Prade, H. (eds.), Fuzzy Sets in Approximate Reasoning and Information Systems, vol. 3, The Handbook of Fuzzy Sets Series, Norwell, MA: Kluwer Academic Publishers.