CROSSOVER IMPROVEMENT FOR THE GENETIC ALGORITHM IN INFORMATION RETRIEVAL

DANA VRAJITORU*

Université de Neuchâtel, Institut interfacultaire d'informatique, Pierre-à-Mazel 7, CH-2000

Neuchâtel, Switzerland

(*Received March 1997; accepted February 1998*)

Abstract - Genetic algorithms (GAs) search for good solutions to a problem by operations inspired from the natural selection of living beings. Among their many uses, we can count information retrieval (IR). In this field, the aim of the GA is to help an IR system to find, in a huge documents text collection, a good reply to a query expressed by the user. The analysis of phenomena seen during the implementation of a GA for IR has brought us to a new crossover operation. This article introduces this new operation and compares it with other learning methods.

1. INTRODUCTION

Inspired by the mechanisms of reproduction of living beings, the genetic algorithms constitutes an interesting paradigm that seems to be able to solve many different problems (Holland, 1975; Goldberg, 1989). This general strategy belongs to the class of probabilistic algorithms that use random choices and behave differently even when applied repeatedly on the same data (Brasard and Bratley, 1994).

Several researchers have used the GA in IR and their results seem to indicate that this algorithm could be efficient. In this vein, the main directions concern modifying the document indexing (Gordon, 1988; Blair, 1990), the clustering problem (Raghavan and Agarval, 1987; Gordon, 1991) and improving query formulation (Yang *et al.*, 1992; Petry *et al*., 1993; Chen, 1995). In order to integrate the GA in our previous research (Vrajitoru, 1997), we have considered Gordon's model.

One major problem encountered by whoever wants to use the GAs to solve a problem is the appropriate choice of the underlying operators and their parameter settings. These algorithms have many different forms and, for each of them, several parameters influence their behavior. Many researchers have studied their choices (Mitchell *et al*., 1991; Spears, 1995) and their results suggest that there is no general best GA, and that each form of GA can be better than others, depending on the particular application it is used for.

In this vein, the present paper tries to improve a very important GA feature, that is the crossover operation. From the invention of the GAs (Holland, 1975), several variations of crossover have been developed (De Jong, 1975; Syswerda, 1989) and various studies have shown that the employed form of crossover can determine the performance of the GA (Spears, 1995). The present research is based on the same idea.

* E-mail: dana.vrajitoru@seco.unine.ch.

Although many researchers are optimistic about applying the GAs to solve IR problems, some of our previous experiments have shown poor results. Looking for an explanation, this paper presents an analysis of the importance of the crossover operation in IR.

Section 2 presents a brief introduction to the GAs in IR. Section 3 describes a theoretical analysis to understand some of the phenomena that can lead a GA to poor results. This analysis also conducted us to design a new crossover operator and an experimental phase has confirmed our theoretical approach. Not only does the new crossover operator perform better than the classical one, but it allows the GA to be competitive with other learning methods in the IR field, as shown in Section 4.

2. THE GA IN INFORMATION RETRIEVAL

This section presents a form of GA and how we used it in our information retrieval research.

2.1. Short description of a GA

The GAs are generally used for optimization problems. Through operations inspired from the natural selection, they search for the best solution to a problem (Goldberg, 1989).

Given a search space E, we must find an element $ind_{opt} \in E$ maximizing a performance mapping f defined on E. The elements of E are called individuals and each of them represents a potential solution to the problem. To apply a GA, the individuals must be coded as a sequence of genes called chromosome. The position of a given gene in the chromosome sequence is called locus.

The GA starts with an initial population containing a number of individuals and representing the generation number 0 ($G_0 = \{ind_1, ind_2, .., ind_{nind}\}$). Given an old generation, a new generation is built from it according to the following steps :

- The *reproduction* step selects *nind* individuals from the old generation, according a better chance to individuals presenting a better performance.
- The *crossover* step groups the chosen individuals in couples, chooses a random position or cross site (*crossSite*), and exchanges the resulting parts of each individual from the couple to form two new individuals or children, as follows:

$$child_1(i) = \begin{cases} parent_1(i) & \text{if } i \le crossSite \\ parent_2(i) & \text{if } i > crossSite \end{cases} \qquad child_2(i) = \begin{cases} parent_2(i) & \text{if } i \le crossSite \\ parent_1(i) & \text{if } i > crossSite \end{cases}$$

- The *mutation* step chooses a random gene and replaces its value with a different one (its opposite if the genes are binary).

The GA consists in building new generations until a stop condition is fulfilled (usually the population convergence) or until a given number of generations is reached.

2.2. Problem coding in information retrieval

To evaluate the GA in information retrieval we have used the CACM collection (3204 documents and 50 queries with known relevance judgments), and the CISI collection (1460 documents and 35 queries).

To modify the document indexing with a GA, a chromosome should contain the document representations. Gordon's model represents an individual as a single document descriptor. The

initial population contains several descriptions of the same document that are meant to compete with each other. To extend the model to a real-scale collection, we considered all the documents as a whole. Thus, in our case, the search space (E) is the set of all possible descriptions of the documents in the collection.

Given a set of terms t_k where k = 1, ..., *termNr* , the genetic representation or descriptor associated to the document d_j, where j = 1, ..., *colSize* has the following form :

$$d_j = \langle t_{1j}, t_{2j}, \ldots, t_{termNr\,j} \rangle$$

In this formula, the gene t_{kj} corresponds to the term t_k in the document d_j and has the value 0 or 1, whether the term is absent or present in the document description.

To include all the documents in the collection, an individual is built by concatenation of document descriptors d_j , for j = 1, 2, ..., *colSize*:

$$\begin{aligned} ind &= \langle d_1, d_2, \ldots, d_{colSize} \rangle \\ &= \langle t_{11}, \ldots t_{termNr1}, t_{12}, \ldots t_{termNr2}, \ldots, t_{1colSize}, \ldots t_{termNr\,colSize} \rangle \end{aligned}$$

The basis for our research is the vector space model and the ntf·nidf indexing (Salton *et al*., 1983; Turtle, 1990). The genes t_{kj} no longer have binary values, but real values from 0 through 1.

The learning scheme developed in this paper is aimed for a transient learning, meaning that the GA is trying to improve the system's performance only for the current query. For the GA, an individual is composed by the terms occurring in the current query (an average of 11.24 terms for the CACM and of 7.43 for the CISI).

We have evaluated the learning scheme in two ways, called *retrospective* and *user*. The *retrospective* evaluation provides the entire set of relevance judgments to the GA from the start. The results are positively biased (too optimistic), but are important because they show the apparent error rate (Efron, 1986; Kulikowski and Weiss, 1991).

To estimate the error rate in a more accurate way, the GA must not know the relevance judgments of the current query from the start. We have used a method consisting in "showing the user" the top 30 documents retrieved by the system for the current query. The GA starts without knowing the existence of any relevant document and modifies this knowledge through feedback. We have marked these evaluations by *user*.

The fitness function in our case is given by the average precision at eleven recall points.

2.3. Starting population

We have built the initial population in two different manners, named *title*, and *query learn*, both using sources of information completing the automatic indexing.

The *title* population uses partial indexing built from the logical sections of the documents:

the complete indexing, using all logical sections

only the title,

only the keywords given by the author (CACM only),

only the CR (Computer Review Abstract, CACM only), and

two individuals whose genes have only the "0" value (CISI only, to have the same population size as for the CACM).

The population called *query learn* uses relevance judgments. It contains the individual built with all logical sections, and seven others, resulting from the division in seven parts of the set { (Q, T), ∀query Q, ∀term T Q}.

To evaluate this population we have used the "leaving-one-out" method (Efron, 1986; Savoy and Vrajitoru, 1996), which consists, in this case, in ignoring the relevance judgments of the current query when constructing the population.

The results of these evaluations are shown in table 1. The baseline represents the vector space model. A difference of 5% (at least) is considered as significant.

Table 1. Results of the GA in 10 generations

		Precision (% change)	
Evaluation	Population	CACM	CISI
Baseline		32.7	19.83
Retrospective	title	37.93 (+15.99%)	21.38 (+7.8%)
	query learn	38.16 (+16.7%)	24.9 (+25.59%)
User	title	37.92 (+15.97%)	20.0 (+0.87%)
	query learn	37.85 (+15.77%)	22.01 (+11.02%)

3. A NEW OPERATOR

This section is dedicated to the analysis of a certain phenomenon concerning the classical crossover operation, and to the description of the new operator.

3.1. Various crossover operations

The crossover operation used so far is the simplest operation of this kind. There are other forms, and this paragraph presents some of them.

The first generalization of the simple crossover is the *n-point* crossover (De Jong, 1975). In this case, we randomly choose a number of sites and apply *n* simple crossover operations on the parents at once.

The *restricted* crossover operator is identical to the simple one, with the difference that the cross point can only be chosen between the first and the last position where the parents' chromosomes are different.

$$minDif \le crossSite \le maxDif, \quad \text{where} \quad parent_1(i) = parent_2 \quad \text{for} \quad 1 \le i < minDif \quad \text{and} \quad maxDif < i \le L-1$$

where L denotes the chromosome length.

The *uniform* crossover operator (Syswerda, 1989) consists in independently choosing, for each locus i from 1 through L - 1, if the parents genes will be swapped or not. This choice depends on a swap probability noted p_{swap} :

$$child_1(i) = \begin{cases} parent_1(i) & \text{if } rand_i \le p_{swap} \\ parent_2(i) & otherwise \end{cases}, \qquad child_2(i) = \begin{cases} parent_2(i) & \text{if } rand_i \le p_{swap} \\ parent_1(i) & otherwise \end{cases}$$

Finally, the *fusion* operator (Beasley and Chu, 1996) produces only one child from two parents. For each gene, the child inherits the value from one or the other of the parents with a probability according to its performance:

$$\forall i, 1 \le i \le L, \quad child(i) = \begin{cases} parent_1(i) \text{ with probability } \dfrac{f(parent_1)}{f(parent_1)+f(parent_2)} \\ parent_2(i) \text{ with probability } \dfrac{f(parent_2)}{f(parent_1)+f(parent_2)} \end{cases}$$

3.2. Crossover analysis

The main idea of the GAs is to simulate the mechanism of natural selection of living beings, which makes the ecosystems develop and become stable. Within this mechanism, the organisms adapt themselves through generations to their environment and to specifics survival tasks. Through rough competition, the best individuals mate to produce descendants that can inherit the parents' skills and even increase them.

Inspired from this natural phenomenon, the purpose of the crossover operation is to create new individuals having, hopefully, greater performance than their parents. In our case, we have noticed that the children often show a performance greater than the worse performance of the parents but smaller than the best value of the parents. The following considerations try to analyze this phenomenon.

If H is a partial individual, let $o(H)$ be the number of locus in H (its length). We consider that the "useful" part of an individual represents its intersection with an optimal individual. In our case, the useful part of an individual contains genes representing the fact that query terms are present in the description of relevant documents or absent from the description of non-relevant documents. It seems natural that if the useful part of an individual increases, its performance should do the same. In information retrieval this hypothesis holds.

For a crossover operation between the parents $parent_1$ and $parent_2$, let us consider the useful parts of these individuals, noted $H_1 \wp parent_1$ and $H_2 \wp parent_2$. In the general case, by definition, the useful part of an individual is not unique. In this case, let ind_{opt} be the individual of maximal performance having the largest intersection with $parent_1 \approx parent_2$. We define H_1 and H_2 as the intersection of each parent with ind_{opt} :

$$H_1 = parent_1 \cap ind_{opt}, \qquad H_2 = parent_2 \cap ind_{opt}$$

We try to form, by one crossover, an individual *child* containing a useful part bigger than both H_1 and H_2. We can assume that $o(H_1) > o(H_2)$ (otherwise we reverse the individuals). The individual ind_{opt} is the best we can get from $parent_1$ and $parent_2$, so we should also consider the useful part of the child according to it. Thus, we hope to obtain, from crossover between $parent_1$ and $parent_2$, an individual *child* containing H_3, so that $o(H_3) > o(H_1)$.

The crossover site (*crossSite*) divides H_1 and H_2 each in two parts that represent a rate of their total length noted as α for H_1 and by β for H_2, where $0 \le \alpha, \beta \le 1$ (see Fig. 1). If we note by I_{first} the interval [1..*crossSite*], we have :

$$\alpha = \frac{o(H_1 \cap I_{first})}{o(H_1)} \qquad \beta = \frac{o(H_2 \cap I_{first})}{o(H_2)} \tag{1}$$

The crossover operates on the parents and produces the children illustrated in Fig. 2. The consistent information in a child is obtained by appending the consistent information (dotted part) from the intervals inherited from each parent.

Then we can compute

$$o(H_3) = \alpha \cdot o(H_1) + (1-\beta) \cdot o(H_2), \qquad o(H_4) = (1-\alpha) \cdot o(H_1) + \beta \cdot o(H_2)$$

As we have mentioned before, the performance of $child_1$ exceeds the performance of the parents if the length of its useful information, $o(H_3)$, is greater than the length of the useful information in the parents. As we know that $o(H_1) > o(H_2)$, this condition becomes :

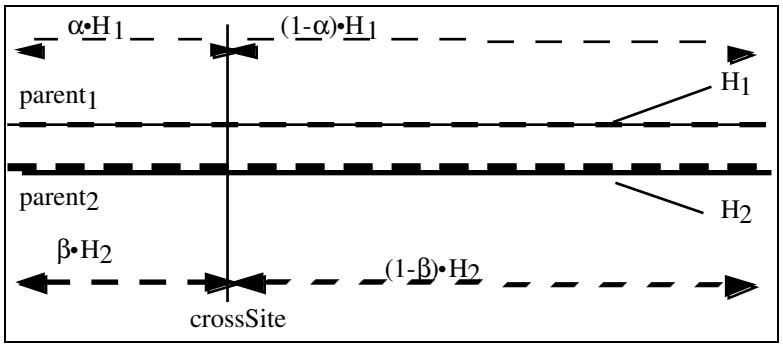

Fig. 1. Meaning of α and β.

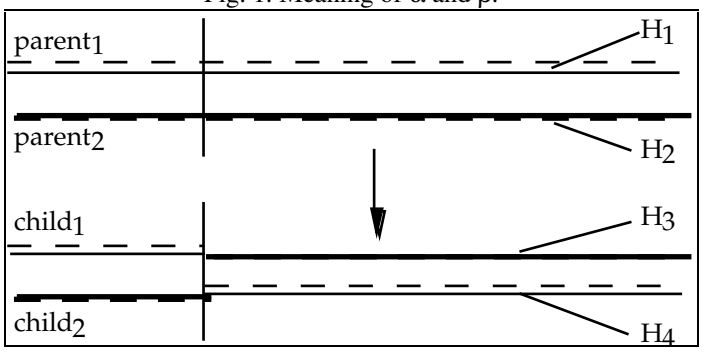

Fig. 2. Crossover result.

$$o(H_3) > o(H_1) \Leftrightarrow \alpha \cdot o(H_1) + (1-\beta) \cdot o(H_2) > o(H_1) \Leftrightarrow (1-\beta) \cdot o(H_2) > (1-\alpha) \cdot o(H_1) \Leftrightarrow \frac{(1-\beta)}{(1-\alpha)} > \frac{o(H_1)}{o(H_2)} \quad (2)$$

So, the initial condition ($o(H_3) > o(H_1)$) is equivalent to the condition in the last line of equation [2]. This means that the chance to obtain an individual of better performance than its parents increases with the difference between α and β. In fact, the ratio between 1 - α and 1 - β must be higher than the rate between the length of the useful information in the parents. We have considered that H_1 belonged to the parent of highest performance, so the rate $o(H_1) / o(H_2)$ is superior to 1. Generally, in our IR application at least, the individuals forming the initial population share similar distributions of the gene values, and we will show that this property determines the ratio (1 - α) / (1 - β) to be very close to 1. In this conditions, the constraint expressed in equation [2] can no longer be fulfilled, and the performance of the population cannot increase.

We will now consider the fact that the optimal individual ind_{opt} includes both H_1 and H_2.

Let I be an interval (a set of loci). We state the *hypothesis* that, because of the uniformity of the population, the length of the intersection of I with each of H_1, H_2, and ind_{opt} is proportional with their length :

$$\frac{o(I \cap H_1)}{o(H_1)} \approx \frac{o(I \cap H_2)}{o(H_2)} \approx \frac{o(I \cap ind_{opt})}{o(ind_{opt})} \quad (3)$$

By the statistical law of big numbers, a large individual size, as in our case, should increase the chances that the hypothesis holds. Equation [3] can be transformed as follows :

$$o(I \cap H_1) \approx o(H_1) \cdot \frac{o(I \cap ind_{opt})}{o(ind_{opt})}, \qquad o(I \cap H_2) \approx o(H_2) \cdot \frac{o(I \cap ind_{opt})}{o(ind_{opt})} \tag{4}$$

If we replace the interval I in equation [4] with I_{first} , we can reconsider the values of α and β according to equation [1]:

$$\alpha = \frac{o(H_1 \cap I_{first})}{o(H_1)} \approx \frac{o(ind_{opt} \cap I_{first})}{o(ind_{opt})}, \quad \beta = \frac{o(H_2 \cap I_{first})}{o(I_{first})} \approx \frac{o(ind_{opt} \cap I_{first})}{o(ind_{opt})}, \quad \alpha \approx \beta \tag{5}$$

Equation [5] indicates that if the population is uniform, the values of α and β are almost equal. As their ratio is close to 1, it is no longer possible to fulfill the condition [2] wherever the cross site may be. Again, this statement implies that the performance cannot be improved. This problem resembles a known problem of afine combinations.

Table 2. Histograms for the CACM collection

Individual	Histogram	#Couples	μ	σ
ind_{opt}	130 135 182 232 451 652 **890** 1303 1374 1923 1020	8292	8.14	2.45
Indexed	15 23 28 54 103 151 **156** 254 241 361 202	1588	8.13	1.93

The hypothesis of uniformity expressed in equation [3] is rather hard to verify. In our case, the useful information in each individual consists in the presence of query terms in the description of relevant documents and the absence of the same search keywords in the description of non-relevant documents. As this last part of the information is to large to significantly influence equation [3], we have considered only the information about relevant documents.

To have an idea about the credibility of the underlying hypothesis of uniformity, we have analyzed the distribution of the query terms in the descriptions of the relevant documents in the researched individual (ind_{opt}) and in the automatically indexed individual for the two collections. The researched individual ind_{opt} is obtained by assembling all the query terms for all the relevant documents and nothing else. We have divided the documents in classes of 300 documents for the CACM and of 150 documents for the CISI. For example, the third class for the CACM collection contains the documents with numbers between #601 and #900. For the CISI collection, the third class contains documents with numbers between #301 and #450.

The number of couples

(*term* Q, relevant document for Q) for any query Q

found in each class are contained in Tables 2 and 3 (values depicted under the label "#couples"). The same information is presented in Fig. 3. The column #couples contains the total number of such couples in each individual and corresponds to $o(H_i)$. We have added to the histograms the average class (μ) and the corresponding standard deviation (σ).

We can notice from Fig. 3 that the distribution of the useful information between classes of both ind_{opt} and indexed individuals are similar, and that the average class and the standard deviation are very close for the two individuals. These remarks confirm the hypothesis expressed in equation [3].

For a better understanding of the meaning of this formula in our case, we have applied it to the seventh class of the CACM collection (document numbers between 1801 and 2100) and to the first class of the CISI collection (document numbers between 1 and 150). Both values are

depicted in bold in Tables 2 and 3. The ratio between the number of couples in these classes and the total number of couples in the individual they come from are very close for each collection :

CACM		CISI	
ind_{opt}	$\frac{890}{8292} = 0.107$	ind_{opt}	$\frac{1218}{9961} = 0.122$
Indexed	$\frac{156}{1588} = 0.098$	*Indexed*	$\frac{316}{2669} = 0.118$

These results suggest that the hypothesis expressed in equation [3] holds and, according to equation [5], the parameters α and β are almost equal. It seems a good explanation for the fact that, in many of the crossover operations that we observed, the children show less performance than the best of the parents, phenomenon that dramatically restricts performance improvement.

Table 3. Histograms for the CISI collection

Individual	Histogram	#Couples	μ	σ
ind_{opt}	**1218** 1179 1148 1244 1401 1101 518 978 587 587	9961	4.84	3.53
Indexed	**316** 295 307 375 349 298 139 280 162 148	2669	4.87	3.54

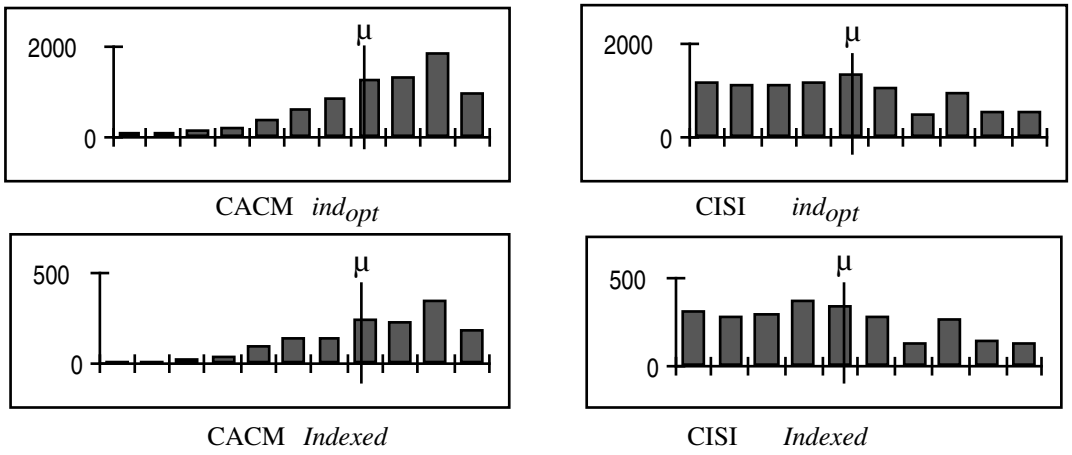

Fig. 3. Graphical presentation of the histograms.

3.3. Description and evaluation of the new operator

The analysis presented in the preceding section has led us to implement a new crossover operator, called *dissociated crossover*. The main idea is to force the parameters α and β to take different values no matter what information the parents might contain. To do this, we have introduced a second cross site.

Let $parent_1$ and $parent_2$ be two individuals and $1 \le crossSite_1 \le crossSite_2 \le L$ two crossover sites. The new individuals $child_1$ and $child_2$ are created in the following manner:

$$child_1(i) = \begin{cases} parent_1(i) & \text{if} \quad i \le crossSite_1 \\ parent_1(i) \text{ or } parent_2(i) & \text{if} \quad crossSite_1 < i \le crossSite_2, \\ parent_2(i) & \text{if} \quad i > crossSite_2 \end{cases}$$

$$child_2(i) = \begin{cases} parent_2(i) & \text{if} \quad i \le crossSite_1 \\ 0 & \text{if} \quad crossSite_1 < i \le crossSite_2 \\ parent_1(i) & \text{if} \quad i > crossSite_1 \end{cases} \tag{6}$$

The difference between the simple two-point and the dissociated crossover operators is depicted in Fig. 4 from which one can see that

- the simple two-point crossover applies the same two simple crossover operations to each parent, but
- the dissociated crossover applies a different simple crossover operator to each parent.

In this case, the question is not "how do we obtain each child", but "what happens to each parent".

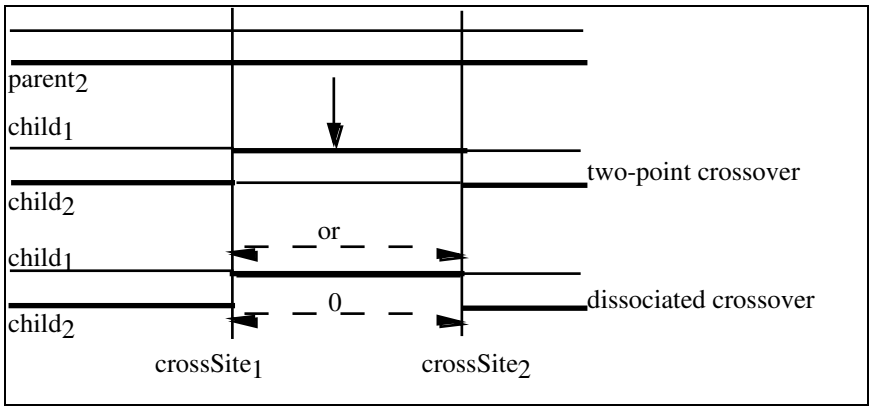

Fig. 4. Dissociated crossover versus two-point crossover.

Table 4. Results of the retrospective transient GA

	Precision (%change)			
	CACM		CISI	
Population	classical	dissociated	classical	dissociated
Query learn	38.16	43.95 (+15.73%)	24.90	25.74 (+3.37%)
Title	37.93	42.01 (+10.76%)	21.38	23.18 (+8.42%)

Table 5. Results of the user transient GA

	Precision (%change)			
	CACM		CISI	
Population	classical	dissociated	classical	dissociated
Query learn	37.85	42.9 (+13.34%)	22.01	24.57 (+11.63%)
Title	37.92	41.49 (+9.41%)	20.0	21.82 (+9.1%)

The difference between the two operators is essential, because the analysis made in Section 3.2 for the simple crossover holds for the n-point crossover ($n > 1$), but does not hold for the dissociated crossover.

The results obtained by the modified GA through various generations show a greater performance diversity. The new method systematically shows better results than the classical

GA and the difference is almost always significant. Tables 4 and 5 present a comparison between the two methods in 10 generations.

To complete the results, Table 6 presents a comparison between the crossover operators, considering the percentage of queries where each algorithm is better and significantly better (difference > 5%) than the other. Thus, the second and third columns treat the new algorithm, the next two treat the classical GA, and the last column shows the percentage of queries where the methods gave the same results. These results confirm the fact that the new operator not only behaves better on the whole, but on a greater number of queries as well.

4. RELEVANCE FEEDBACK

The research in this article uses a transient approach to the GA, as we have specified it in the first section. To compare the GA with a classical transient learning algorithm in information retrieval, we have tested a form of relevance feedback under the same experimental conditions. This chapter presents the method and compares it with the GA.

Many authors have used this simple and efficient method and many variants of it exist. We can find some of them in (Salton and Buckley 1990; Dillon and Desper, 1980). The general idea is to modify the query according to the relevance judgments given by the user, in order to modify the request and to obtain better search results. We have chosen the relevance feedback variant showing the best results in (Salton and Buckley, 1990), that is the *dec-hi* (Ide, 1971).

The method consists in showing the user a chosen number of documents, classified by the system on the top of the list, and to make him judge them. On this basis, the query is modified by including or enhancing the terms appearing in the relevant documents and removing the terms appearing in the first top non-relevant document.

Table 6. Comparison of the operators by query percentage

Collection	Dissociated	Significant	Classical	Significant	Equality
CACM	60.33%	47.67%	9.67%	5.33%	30%
CISI	70%	46.67%	20.95%	10.48%	9.05%

Table 7. Residual evaluations for the CACM collection

	Precision (%change)				
Algorithm	5 docs	10 docs	15 docs	20 docs	30 docs
GA - query learn dissociated	32.70-41.57 (+27.67%)	32.70-44.78 (+37.53%)	32.70-44.07 (+35.36%)	32.70-44.64 (+37.1%)	32.70-42.9 (+31.2%)
Rel FB - user	19.34-25.95 (+34.19%)	16.31-24.24 (+48.62%)	15.57-23.83 (+53.08%)	13.48-22.1 (+64.01%)	11.70-21.87 (+86.95%)

Table 8. Residual evaluations for the CISI collection

	Precision (%change)				
Algorithm	5 docs	10 docs	15 docs	20 docs	30 docs
GA - query learn dissociated	19.83-22.55 (+12.08%)	19.83-24.81 (+23.3%)	19.83-25.44 (+26.46%)	19.83-25.24 (+25.44%)	19.83-24.57 (+23.9%)
Rel FB - user	17.69-22.31 (+26.16%)	16.70-22.71 (+36.02%)	16.34-23.24 (+42.28%)	14.62-22.83 (+56.14%)	13.07-23.73 (+81.62%)

The query is modified according to the following equation :

$$q' = q + \sum_{all\ rel} d_i - d_{top\ non\ rel}$$

in which *q'* is the new query, *q* is the previous query and d_i document vectors.

If the evaluation of the modified request includes the documents already seen, we can call this evaluation a retrospective one, and it leads to a biased performance measure. A second evaluation method, called *residual*, removes from the final retrieved list all the documents seen by the user. Tables 7 and 8 show the results of this evaluation compared with the GA (10 generations), by variation of the number of documents seen by the user (notation: 5 through 30 *docs*).

As the two methods do not have the same performance baseline, we cannot directly compare their results. The only comparison criteria can be the percentage of change (in parenthesis in the tables) from the baseline. According to it, the relevance feedback seems to perform better than the GA. Moreover, the former method is faster, easier to implement, and easier to understand.

We also know that it is easier to increase the performance from a low baseline. It is the case for the relevance feedback, and this fact moderates our optimism about it. As an advantage of the GA, we can cite its probabilistic feature that assures a different result at each run even on the same query.

5. CONCLUSION

The goal of this article is to introduce a new crossover operator for the GA used in IR. The analysis presented in the third section shows the origin of the new operator, and the results, compared to the classical GA, indicate that the crossover operator can be improved.

Thus, the new operator shows significantly and systematically better results than the classical one. This fact indicates that the new operator is well adapted for our research domain (IR) and encourages us to continue this research in other domains.

A comparison between our application of the GA and the method of the relevance feedback shows that, even if the GA is less efficient than more direct methods, it still has its advantages and will probably continue to be studied in the future.

Acknowledgments - This research was supported by the SNSF (Swiss National Science Foundation) under grant 20-43'217.95.

REFERENCES

Beasley, J. E. & Chu, P. C. (1996). A Genetic Algorithm for the Set Covering Problem. *European Journal of Operational Research*, *94*, 392-404.

Blair, D.C. (1990). *Language and Representation in Information Retrieval*. Amsterdam: Elsevier.

Brassard, G. & Bratley, P. (1994). *Fundamentals of Algorithmics*. Prentice Hall.

Chen, H. (1995). Machine learning for information retrieval: Neural networks, symbolic learning, and genetic algorithms. *Journal of the American Society for Information Science, 46*(3), 194-216.

De Jong, K. A. (1975). An Analysis of the Behavior of a Class of Genetic Adaptive Systems. (Doctoral dissertation, University of Michigan). *Dissertation Abstracts International*, *36*(10), 5140B.

Dillon, M., & Desper, J. (1980). Automatic relevance feedback in Boolean retrieval systems. *Journal of Documentation*, *36*, 197-208.

Efron, B. (1986). How Biased Is the Apparent Error Rate of a Prediction Rule. *Journal of the American Statistical Association, 81* (394), 461-470.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley.

Gordon, M. (1988). Probabilistic and genetic algorithms for document retrieval. *Communications of the ACM, 31*(10), 1208-1218.

Gordon, M. (1991). User-Based Document Clustering by Redescribing Subject Descriptions with a Genetic Algorithm. *Journal of the American Society For Information Science, 42*(5), 311-322.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor: Univ. of Michigan Press.

Ide, E. (1971). New experiments in relevance feedback. In *The Smart system - experiments in automatic document processing*, 373-393, Englewood Cliffs, NJ: Prentice Hall Inc.

Kulikowski, A.C. & Weiss, M.S. (1991). *Computer systems that learn.* San Mateo, CA: Morgan Kaufmann.

Mitchell, M., Forrest, S. & Holland, J.H. (1991). The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Toward a practice of autonomous systems: proceeding of the first european conference on artificial life,* Cambridge (MA): The MIT Press.

Petry, F., Buckles, B., Prabhu, D., & Kraft, D. (1993). Fuzzy information retrieval using genetic algorithms and relevance feedback. In *Proceeding of the ASIS annual meeting*, 122-125.

Raghavan, V.V. & Agarwal, B. (1987). Optimal determination of user-oriented clusters : An application for the reproductive plan. In *Proceedings of the second conference on genetic algorithms and their applications*, Hillsdale, NJ, (pp. 241-246).

Salton, G., & Buckley, C. (1990). Improving performance by relevance feedback. *Journal of the American Society for Information Science*, *41*(4), 288-297.

Salton, G., Fox, E., & Wu, U. (1983). Extended Boolean information retrieval. *Communications of the ACM*, *26*(12), 1022-1036.

Savoy, J. & Vrajitoru, D. (1996). *Evaluation of Learning Schemes Used in Information Retrieval.* Technical Report CR-I-95-02, Université de Neuchâtel, Faculté de droit et des Sciences Économiques.

Spears, W. (1995). Adapting crossover in evolutionary algorithms. *Proceedings of the fourth annual conference on evolutionary programming.*

Syswerda, G. (1989). Uniform crossover in genetic algorithms. In J. D. Schaffer (Ed.), *Proceedings of the third international conference on genetic algorithms*, San Mateo (CA): Morgan Kaufmann Publishers.

Turtle, H. (1990). *Inference networks for document retrieval.* Doctoral Dissertation, Computer and Information Science Department, University of Massachusetts. Technical Report COINS Report 90-92, October 1990, ACM-TOIS.

Vrajitoru, D. (1997). *Apprentissage en recherche d'informartions.* Doctoral thesis, University of Neuchâtel, Faculty of Science.

Yang, J.-J., Korfhage, R.R., & Rasmussen, E. (1992). Query improvement in information retrieval using genetic algorithms. *Proceedings of TREC'1*, NIST, Gaitherburgs (MD), (pp. 31-58).

Information Processing & Management, Vol. 34, No. 4, pp. 405-415, 1998.