Query reformulation using WordNet and genetic algorithm

Conference Paper · March 2017

DOI: 10.1109/NTICT.2017.7976138

3 authors:

Belal Al-Khateeb
University of Anbar

Ali J. Al-Kubaisi
University of Fallujah

Sufyan T. Faraj
University of Anbar

Query Reformulation Using WordNet and Genetic Algorithm

Belal Al-Khateeb
Computer Science Dept.
College of CS and IT
University of Anbar, Ramadi, Iraq
(belal@computer-college.org)

Ali J. Al-Kubaisi
Computer center and Information Technology
University of Fallujah, Fallujah, Iraq
(alijamal530@gmail.com)

Sufyan T. Al-Janabi[1,2]
[1]College of CS and IT
University of Anbar, Ramadi, Iraq
[2]CST, University of Human Development
Sulaimaniya, KRG-Iraq
(saljanabi@fulbrightmail.org)

***Abstract*— Search on the web is a delay process and it can be hard task especially for beginners when they attempt to use a keyword query language. Beginner (inexpert) searchers commonly attempt to find information with ambiguous queries. These ambiguous queries make the search engine returns irrelevant results. This work aims to get more relevant pages to query through query reformulation and expanding search space. The proposed system has three basic parts WordNet, Google search engine and Genetic Algorithm. Every part has a special task. The system uses WordNet to remove ambiguity from queries by displaying the meaning of every keyword in user query and selecting the proper meaning for keywords. The system obtains synonym for every keyword from WordNet and generates query list. Genetic algorithm is used to create generation for every query in query list. Every query in system is navigated using Google search engine to obtain results from group of documents on the Web. The system has been tested on number of ambiguous queries and it has obtained more relevant URL to user query especially when the query has one keyword. The results are promising and therefore open further research directions.**

***Index Terms*— Search Engine, SE, Artificial Intelligence, AI, WordNet, Intelligent Search Algorithms, Query Reformulation.**

I. INTRODUCTION

FOR thousands of years, people have understood the importance of archiving and finding information. The first libraries created in Sumerian civilization around 3000 BC to keep written record and written literature, when archiving was with cuneiform inscriptions. The Sumerian understood they must index the archives to use the Sumerian designed specific region to store clay tablets archives efficiently. They succeeded to develop a special classification to identify each tablet and its content [1] [2]. Over the centuries, the need to store and retrieve written information became gradually more important, mainly after invention of computers enabling the possibility of storing large amount of information. People realized that the automatic archiving and retrieve of information is very import. The birth of the idea of automatic access to large amounts of stored information was the article of the Vannevar Bush in 1945 with the article title "AS We May Think" [1] [3].

IR is the activity of finding stored information related to an information need from a collection of stored information [5]. The main task of the IR system is finding relevant information to the user's query [6].

The infrastructure of Intelligent Search Engine (ISE) contains various systems, which are search engine, semantic web, and IR system. Semantic web can solve the first problem in web with semantic annotations to produce intelligent and meaningful information by using query interface mechanism and ontology's. The Semantic web would require solving extraordinarily difficult problems in the areas of knowledge representation, and natural language understanding. ISE is the technologies of semantic web and search engine to improve the search results gained by current search engines and evolves to next generation of search engines built on Semantic Web [7] [8].

II. GENETIC ALGORITHMS IN INFORMATION RETRIVAL

Genetic Algorithms (GAs) were formally presented in the United States in the 1970s by John Holland at University of Michigan. Genetic algorithm is a one of the search approaches in the artificial intelligence field centered on biological principles. It is a common method that can be applied to many problems. It works on two basic fundamentals which are natural selection and natural genetics [18] [19]. In designing of GA, there are three important issues that must be taken into account. Firstly, coding the problem solutions. Secondly, finding a fitness function that can optimize the performance and finally, the set of parameters including the population size, crossover and mutation [20] [21]. The genetic algorithm state in following steps [19]:

a. Initialization: Commonly the initial population is created randomly.
b. Evaluation: Calculates the fitness of the current generation.
c. Selection: Selecting of two parents which are used to generate new generation. The main idea of selection is to select the best solutions to worse ones.
d. Crossover: Combine two or more parental solution used to generate new possibly better solution. There are many methods of crossover among of them are one-point, two-point, uniform, order-based and etc.

e. Mutation: The mutation operates on local solutions. So it randomly modifies the solution.
f. Replacement: Replaces the current generation with new generation.
g. Repeat steps b–f until the final condition is achieved.

Getting relevant documents from huge documents collection is a hard process. GAs they can be used to search the huge document search space. GAs are strong and effective search and especially for large search space. GAs are desirable for information retrieval because of their strength and quick search abilities [20].

Traditional information retrieval models use single query but GA based techniques use a population of queries rather than a single query. Query and documents are denoted as chromosomes. The first step is creating the initial population of documents. The second step is evaluating the population by computing the fitness for each chromosome. The document is considered as relevant if there are a matching between the query chromosome and the document chromosome. The query stays in reformulated process until a relevant document is retrieved [20] [22].

III. WORDNET

The "WordNet" was firstly known in 1985. A group of psychologists and linguists from Princeton University discussed and improved a new subject which called the "Lexical Database". This subject also called “electronic dictionary”, “semantic network”, etc. [23]. The reason behind the invention of the "WordNet" is to combine between the benefits of dictionary and Thesaurus. The basic idea of how to organize "WordNet", is based on sorting depending on the meaning. The main groups for meanings of most frequent English words are four categories which are "Nouns, Verbs, Adjectives, and Adverbs". Another reason behind this invention is that to support automatic text analysis and artificial intelligence applications. So, "WordNet" can be utilized by humans and machines. One of the benefits of the WordNet is its online availability [24]. The WordNet divides the lexicon into five categories: nouns, verbs, adjectives, adverbs, and function words. Explicitly, this is what distinguishes it from a standard dictionary. Indeed, WordNet includes only nouns, verbs, adjectives, and adverbs [17]. There are many relations in WordNet such as [17] [23] [25]:

a. Synonymy: The basic structure of WordNet is the relation of Synonymy. 'Synonymy' means a lexical relation between word forms. Two expressions are called synonymous if the substitution of one for the other never changes the truth value of a sentence in which the substitution is made.
b. Hyponymy: has more than relational names, like “is a” relation, "type of”, or “kind of” relation. Example: Apple: fruit Apple is a fruit. There is a semantic relationship between word meanings called "Hyponymy".
c. Meronymy: Another concept represented by the synset {x, x¢, . . .} is said to be a meronym of the concept represented by the synset {y, y¢, . . .} if native speakers of English accept sentences constructed from such frames as A y has an x (as a part) or An x is a part of y.
d. Antonym: The antonym of a word x is sometimes not-x, but not always. For example, rich and poor are antonyms, but to say that someone is not rich does not imply that they must be poor; many people consider themselves neither rich nor poor.

IV. RELETED WORK

There are many literatures used different techniques for implementing Genetic Algorithm (GA) or WordNet in search engine. Here, some earlier studies in GA in search engine and some other studies in WordNet which are related to this work.

M. Gordon (1988) proposed a GA to extract the document descriptions. He used fixed length for each description and chose binary coding scheme. The genetic population is collected of various descriptions for the same document. The fitness function is computed using Jaccard similarity between the existing document description and each of the queries, and then calculating the average adaptation values of the description to the set of relevant and non-relevant queries [9].

Vrajitoru (1998) proposed a GA with the vector space model. The main disadvantage of this proposal is that the fitness function only considers single query, and thus the document descriptions are modified to match with this one query and not with a group of queries as in Gordon's model [10].

Pathak et al. (2000) suggested a new weighted matching function, which is the linear combination of different existing similarity functions. They used Gaussian noise mutation, read coding and two point crossover. The proposed GA admirably was with the simulated document collection and the Cranfield document collection. It is additionally important to test this algorithm on different document collections to see how it performs with scaling both in size of the database and in the features available [11].

D. Husek et al (2005) used GA to in information retrieval to optimize Boolean query. The query represented as tree (terms in query as leaves and Boolean operators as nodes). The query with fewer operators is better than query with more operators and the same values of precision and recall and it is very hard for a user to formulate a good query [12].

J. Nemrava (2006) used WordNet Glosses to refine Google queries. He designated some techniques how to arrange returned web sites into appropriate synonym classes using WordNet. He tested a set of 50 proper nouns from several different areas. He got precision 62 percent from these 50 test concepts with 92 synonyms in total. The main problem with proposed system is the response time. The typical time to determine one synonymic class is around 50 seconds with typical 20 Google questions for every one equivalent word class. [13].

José R. (2007) used GA for query reformulation. He used fix-length binary strings where each position corresponds to a candidate query term, roulette wheel selection, one-point crossover operator and random mutation [14].

Abdelmgeid A. Aly (2007) proposed adaptive method using GA to modify user's queries, established on relevance judgments. This algorithm was adapted for the three well-known documents collections (CISI, NLP and CACM). The algorithm displays the special effects of applying GA to increase the effectiveness of queries in IR systems. The proposed GA approach gives better results than classical IR system when tested [15].

Ahmed A. A. Radwan et al (2008) proposed a new fitness function for estimated information retrieval which is very fast and very flexible, than cosine similarity fitness function. They used three well-known documents collections [16].

Ashish K. and Nitin C. (2014) proposed a hybrid strategy for improving the search engine results via document clustering, genetic algorithm and Query Recommendation to furnish the user with the best significant pages to the search query. The proposed system begins with query recommendation, genetic algorithm are applied to consequent pages from query recommendation to send most relevant pages to user at smallest time [17].

V. PROPOSED SYSTEM MODEL

The proposed system takes query input into the preprocessing stage, use WordNet. Next, the system uses Google Search engine and GA, as shown in Fig. 1, the processing stage have three steps: tokenization, removing stop words and lemmatization. Each term is separated from other in tokenization process. After tokenization is done, the stop words are removed from query then each term is returned to base or dictionary form through lemmatization process. Thereafter, WordNet is used to display the synonymy, polysemy and glosses for each term. After the user selects the proper meaning, the Google search engine is used to obtain results. Next, GA runs to select better solution.

The system model uses Google Search engine to navigate the query. After query navigated, we parse the URL pages in Google result. Through parsing the Google results, we can obtain useful information that can be used in ranking the results in GA such as the title, description of the page, URL of the page, order of the page in Google results, etc. Parse of Google results stated in algorithm 1.

GA represents the query and documents as chromosome. The result of the navigate query in Google search engine will be population. The proposed system generates all possible queries for original query (for example if we have query such as " A B C", and every term in query have list of synonym, tem "A" have three synonyms, term "B" have two synonyms and term "C" have four synonyms then all possible queries are twenty four queries (See Fig. 2). GA generates the initial population for the query chromosome. Then the fitness will be computed for every URL by matching the query and document chromosome. The proposed system model uses GA to refine results and reformulation the query. Google Search engine is used to navigate the query. The steps of query reformulation using WordNet and GA explained in Algorithm 2.

VI. FITNESS FUNCTION

Fitness function is the basic part of GA. Good results come from proper selection of the fitness function. The proposed model parses every URL page and uses the meta-data of URL page. The system compute the fitness as stated in Algorithm 3. The fitness function has four factors used to evaluate the chromosome. The first factor is the order of the URL in result of Google. The Second factor is similarity between the gloss of the keywords description and the URL description. The third factor is similarity between the query and the URL description. The last factor is the similarity between the query and the URL title. These factors have different degree of importance. The system test 84 possible probabilities for A, B, C and D have total sum one. The system tests different queries with all possible situations of factors A, B, C, and D. When using probability (0.1, 0.2, 0.3, 0.4) we obtain best results relevant to user query because we obtain higher importance (0.4) to Similarity (Title of URL, Query), (0.3) to Similarity (Query, Description of URL), (0.2) to Similarity (Glosses of keywords, Description of URL) and lower importance (0.1) to order of URL in results. For example if we have query "Java" with meaning ("a beverage consisting of an infusion of ground coffee beans"). The fitness growths especially when the probability of order in Google increased. But this increasing must be controlled under percentage of relevant of results to user query. In Table 1 we note the probability A=0.1, B=0.2, C=0.3 and D=0.4 gave best results.

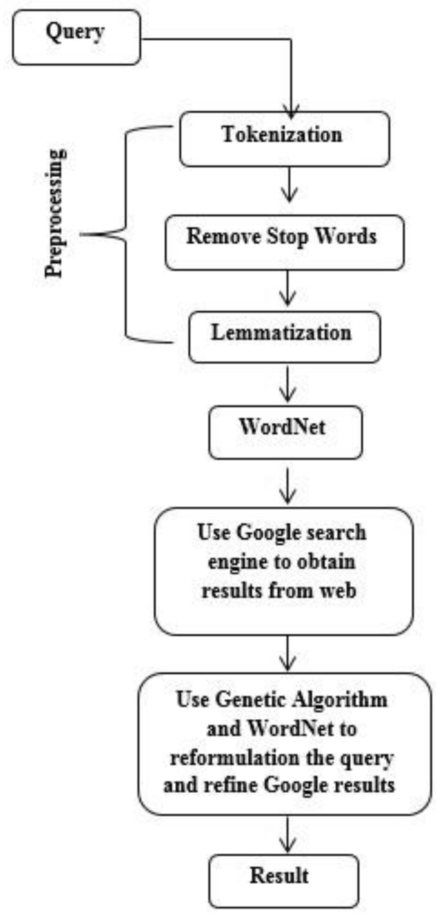

Fig. 1 Simplified System Model.

Algorithm1. Parse of Google Result

1- For every chromosome in generation (web page in Google result) do the following:
   - Access to the Meta-data in the web page.
   - Obtain the title of the web page.
   - Obtain the description of the web page.
   - Obtain the URL of the web page.
   - Obtain the Order of the web page in Google Results.
2- Take the result of the step 1 and used it in GA to find the fitness of the page according to certain considerations.

Input Query= "A B C"

Then if

- Term "A" have three synonyms
- Term "B" have two synonyms
- Term "C" have four synonyms

The all possible queries are:

1	A1 B1 C1	**7**	A1 B2 C3	**13**	A2 B2 C1	**19**	A3 B1 C3
2	A1 B1 C2	**8**	A1 B2 C4	**14**	A2 B2 C2	**20**	A3 B1 C4
3	A1 B1 C3	**9**	A2 B1 C1	**15**	A2 B2 C3	**21**	A3 B2 C1
4	A1 B1 C4	**10**	A2 B1 C2	**16**	A2 B2 C4	**22**	A3 B2 C2
5	A1 B2 C1	**11**	A2 B1 C3	**17**	A3 B1 C1	**23**	A3 B2 C3
6	A1 B2 C2	**12**	A2 B1 C4	**18**	A3 B1 C2	**24**	A3 B2 C4

Fig. 2 Example of all possible queries to certain query.

During this choice the system took highest precision, because the probability of similarity between the user query and title of page higher than others. However, if the system chooses the probability A=0.7, B=0.1, C=0.1 and D=0.1 then the highest fitness obtained because the probability of order of URL in Google results higher than others. But the system returns minimal value of precision due to the use ambiguous queries, and it depended on order of URL in Google results (see Fig 3 and Fig 4).

VII. Testing the Viability of the System

Our tests have been designed for testing how much the WordNet and GA can improve the results of the search by retrieving more relevant documents. The results of Google and the results of the system after applying the WordNet and GA evaluated using Precision. Precision describes the amount of the valid information in the search results, which reflects the helpfulness for users. In other words, Precision is the fraction of the documents retrieved that are relevant to the information need. We test number of ambiguous queries with one keyword. After the system shows the glosses for keyword we choose the meaning for the keyword. The system searches using Google to search for all possible queries for original query. We obtain more relevant URL to user query after applying the system because the system expands the search space and use new query to obtain best percentage of precision. In query "Java", G.A. applying two generations to introduce the final result. In query "Pluto", G.A. applying four generations to create the final result. In query "Boot", G.A. applying seven generations to create final result. We note the precision of ambiguous queries before applying the system is low. If the query has higher degree of ambiguity then the precision is low before applying the system. This system designed to help the non-professional user when use ambiguous queries but, for worst case if the user attempts to find result with clear queries then the system returns result such as in result from Google. For example, when the user has query such as "How to visit Java?". This query is clear and the Google return result about island Java.

Algorithm 2. Query Reformulation using WordNet and GA

1- Take the query ***q*** from user.
2- Pre-process ***q*** by doing the following:
   - Tokenize ***q*** to ***L***ist of ***K***eywords (tokens) ***LK***.
   - Eliminate stop words from ***LK***.
   - Stemming every keyword in ***LK***.
3- Display the glosses and synonym for every keyword in ***LK*** using WordNet.
4- Take the gloss and synonym for every keyword in ***LK*** after selecting the correct meaning of a user.
5- Generate ***L***ist of all possible ***Q***ueries ***LQ*** (see figure 2).
6- For ***i***=1 to length of ***LQ*** do the following:
   - Create Generation ***i*** (***Gen[i]***) by navigate the q_i using Google search engine. Where every URL in results of Google represented chromosome in ***Gen[i]***.
   - For every chromosome in ***Gen[i]*** do the following:
     • Parse every web page to obtain Title, Description and Order of web page in Google result using Algorithm 1.
     • Compute fitness for every chromosome in ***Gen[i]*** using Algorithm 3.
     • Select best chromosomes from ***Gen[i]*** and store it in final results.
7- Return the final results to end user.

Algorithm 3. Fitness Function for proposed system Algorithm

1- For every chromosome (web page) in generation do the following :
- Compute ***O***, where ***O*** is order of web page in Google result.
- Compute ***S1***, where ***S1*** is similarity between gloss of keywords and description of web page and can be compute it using algorithm 4.
- Compute ***S2***, where ***S2*** is similarity between query and description of web page and can computed using algorithm 4.
- Compute ***S3***, where ***S3*** is similarity between query and title of web page and can computed using algorithm 4.
- The Fitness of chromosome is:
  - Fitness=A****O*** + B****S1*** + C****S2*** + D****S3*** ,
    Where A, B, C and D are the degree of importance.

2- Return the fitness for every chromosome.

Algorithm 4. Jaccard score

1- ***Str1*** and ***Str2*** is two strings.
2- Compute ***Inter***, where ***Inter*** is intersection between ***Str1*** and ***Str2***.
3- Computer ***Uni***, where ***Uni*** is union between ***Str1*** and ***Str2***.
4- Compute Jaccard score ***J***, where J is ***length of Inter/ length of Uni***.
5- Return J.

Table 1. Precision for all possible factors A, B, C and D used in Fitness function for query "Java", where F is a Fitness and P is Precision.

No.	A	B	C	D	F	P
1	0.1	0.1	0.1	0.7	0.308	50%
2	0.1	0.1	0.2	0.6	0.308	48%
3	0.1	0.1	0.3	0.5	0.308	49%
4	0.1	0.1	0.4	0.4	0.308	52%
5	0.1	0.1	0.5	0.3	0.308	49%
6	0.1	0.1	0.6	0.2	0.308	42%
7	0.1	0.1	0.7	0.1	0.308	40%
8	0.1	0.2	0.1	0.6	0.293	46%
9	0.1	0.2	0.2	0.5	0.293	50%
10	**0.1**	**0.2**	**0.3**	**0.4**	**0.293**	**72%**
11	0.1	0.2	0.4	0.3	0.293	70%
12	0.1	0.2	0.5	0.2	0.293	65%
13	0.1	0.2	0.6	0.1	0.293	52%
14	0.1	0.3	0.1	0.5	0.278	39%
15	0.1	0.3	0.2	0.4	0.278	53%
16	0.1	0.3	0.3	0.3	0.278	45%
17	0.1	0.3	0.4	0.2	0.278	48%
18	0.1	0.3	0.5	0.1	0.278	41%
19	0.1	0.4	0.1	0.4	0.263	46%
20	0.1	0.4	0.2	0.3	0.263	32%
21	0.1	0.4	0.3	0.2	0.263	31%
22	0.1	0.4	0.4	0.1	0.263	30%
23	0.1	0.5	0.1	0.3	0.248	45%
24	0.1	0.5	0.2	0.2	0.248	47%
25	0.1	0.5	0.3	0.1	0.248	37%
26	0.1	0.6	0.1	0.2	0.233	30%
27	0.1	0.6	0.2	0.1	0.233	32%
28	0.1	0.7	0.1	0.1	0.218	29%
29	0.2	0.1	0.1	0.6	0.381	36%
30	0.2	0.1	0.2	0.5	0.381	38%
31	0.2	0.1	0.3	0.4	0.381	39%
32	0.2	0.1	0.4	0.3	0.381	46%
33	0.2	0.1	0.5	0.2	0.381	40%
34	0.2	0.1	0.6	0.1	0.381	39%
35	0.2	0.2	0.1	0.5	0.366	45%
36	0.2	0.2	0.2	0.4	0.366	47%
37	0.2	0.2	0.3	0.3	0.366	50%
38	0.2	0.2	0.4	0.2	0.366	49%
39	0.2	0.2	0.5	0.1	0.366	41%
40	0.2	0.3	0.1	0.4	0.351	53%
41	0.2	0.3	0.2	0.3	0.351	50%
42	0.2	0.3	0.3	0.2	0.351	48%
43	0.2	0.3	0.4	0.1	0.351	42%
44	0.2	0.4	0.1	0.3	0.336	46%
45	0.2	0.4	0.2	0.2	0.336	44%
46	0.2	0.4	0.3	0.1	0.336	40%
47	0.2	0.5	0.1	0.2	0.321	41%
48	0.2	0.5	0.2	0.1	0.321	37%
49	0.2	0.6	0.1	0.1	0.306	34%
50	0.3	0.1	0.1	0.5	0.454	40%
51	0.3	0.1	0.2	0.4	0.454	39%
52	0.3	0.1	0.3	0.3	0.454	41%
53	0.3	0.1	0.4	0.2	0.454	35%
54	0.3	0.1	0.5	0.1	0.454	30%
55	0.3	0.2	0.1	0.4	0.439	37%
56	0.3	0.2	0.2	0.3	0.439	36%
57	0.3	0.2	0.3	0.2	0.439	34%
58	0.3	0.2	0.4	0.1	0.439	31%
59	0.3	0.3	0.1	0.3	0.424	39%
60	0.3	0.3	0.2	0.2	0.424	37%
61	0.3	0.3	0.3	0.1	0.424	36%
62	0.3	0.4	0.1	0.2	0.409	37%
63	0.3	0.4	0.2	0.1	0.409	31%
64	0.3	0.5	0.1	0.1	0.394	30%
65	0.4	0.1	0.1	0.4	0.527	40%
66	0.4	0.1	0.2	0.3	0.527	42%
67	0.4	0.1	0.3	0.2	0.527	39%
68	0.4	0.1	0.4	0.1	0.527	37%
69	0.4	0.2	0.1	0.3	0.512	40%
70	0.4	0.2	0.2	0.2	0.512	39%
71	0.4	0.2	0.3	0.1	0.512	31%
72	0.4	0.3	0.1	0.2	0.497	34%
73	0.4	0.3	0.2	0.1	0.497	35%
74	0.4	0.4	0.1	0.1	0.482	34%
75	0.5	0.1	0.1	0.3	0.6	36%
76	0.5	0.1	0.2	0.2	0.6	38%
77	0.5	0.1	0.3	0.1	0.6	31%
78	0.5	0.2	0.1	0.2	0.585	34%
79	0.5	0.2	0.2	0.1	0.585	35%
80	0.5	0.3	0.1	0.1	0.57	38%
81	0.6	0.1	0.1	0.2	0.673	39%
82	0.6	0.1	0.2	0.1	0.673	34%
83	0.6	0.2	0.1	0.1	0.658	30%
84	0.7	0.1	0.1	0.1	0.746	28%

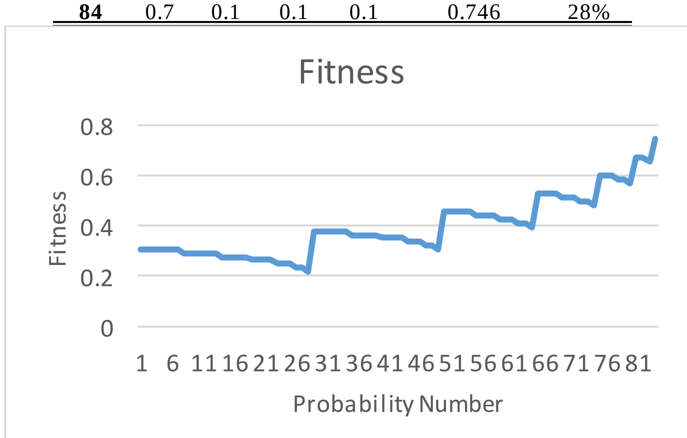

Fig 3. Fitness for all possible factors A, B, C and D used in Fitness function for query "Java".

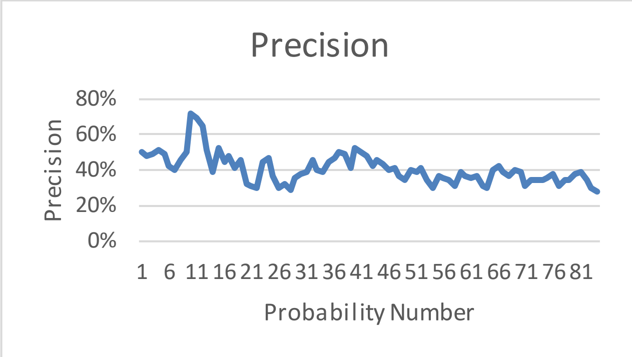

Fig 4. Precision for all possible factors A, B, C and D used in Fitness function for query "Java".

VIII. Conclusions

In this paper, WordNet and GA used to return more relevant results to user query. The implemented system was tested on number of queries. According to the obtained result we can note Using WordNet helps the user to select the proper meaning of the keyword especially, when the keyword has different meanings, Synonym of the keyword helps the system to expand the search space and increases the degree of the precision, If every keyword in query have large number of synonymy, this leads to more generations in GA, highest degree of precision and less response time, GA evaluates the result from Google and selects best result from every generation to generate the final result, Precision of selected query after applied the system was very good, because the system formulates the query depending on user selection from many different meaning, Queries that has one keyword more ambiguous than other queries that have two or more keywords. So, the system produces very good results when queries be more ambiguity, The proposed system efficiency increases, if there is ambiguous in queries and if the query is clear then the system returns results same as result in Google. Some ideas for future extensions of the work developed in this work such as: Generalize and specialize the search space using other relations in WordNet such as Hyponymy and Meronymy, other AI algorithms can be used such as simulated annealing to find optimal solution and comparing it to GA in order to find a best method for finding the results, the system may be implemented on Arabic language using Arabic WordNet with GA to generate right solution or translator in Google search engine may be used to clear the keywords in query and obtain the Synonymy for each keyword rather than using the WordNet.

References

1. Bush, V. 1945. "*As We May Think*", the Atlantic,. Reprinted in Life magazine.
2. Singhal, A. 2001. "*Modern Information Retrieval: A Brief Overview*", IEEE Data Engineering Bulletin, [vol. 24, pp. 35-43].
3. Dalby, A. 1986. "*The Sumerian Catalogs*", the Journal of Library History, [Vol. 21, No. 3, pp. 475-487].
4. Gordon, M. 1988. Probabilistic and genetic algorithms for document retrieval, Communications of the ACM 31(10) 1208–1218.
5. Vrajitoru, D. 1998. Crossover improvement for the genetic algorithm in information retrieval, Information Processing and Management 34(4) 405–415.
6. Chen, H., Shankaranarayanan, G., and She, L. 1998. "*A machine learning approach to inductive query by examples: an experiment using relevance feedback, ID3, genetic algorithms, and simulated annealing*", Journal of the American Society for Information Science, [Vol. 49, No. 8, pp. 693–705].
7. Pathak, P. , Gordon, M., Fan, W., 2000. Effective information retrieval using genetic algorithms based matching functions adaptation, in, Proc. 33rd Hawaii *International Conference on Science (HICS)*, Hawaii, USA.
8. Húsek, D. , Snasel, V. ,Owais, S and Krömer, P. 2005. "Using Genetic Algorithms for Boolean Queries Optimization", *9th IASTED International Conference on Internet and Multimedia Systems and Application*, IEEE, [pp. 178-183].
9. Nemrava, J. 2006. "*Using WordNet glosses to refine Google queries*" In Proc. of the Dateso 2006 Workshop. VSB–Technical University of Ostrava, Dept. of Computer Science, pp. 85-94.
10. Pérez-Agüera, J. 2007. "*Using genetic algorithms for query reformulation*", BCS IRSG Symposium: Future Directions in Information Access.
11. Abdelmgeid, A., 2007. "*Applying genetic algorithm in query improvement problem*", International journal "Information Technologies and Knowledge" Vol 1: 309 - 316.
12. Radwan, A., Latef, B., Ali, A. and Sadek, O. 2008. "*Using genetic algorithm to improve information retrieval systems.*" International Scholarly and Scientific Research & Innovation 2(5):1544-1550.
13. Kushwaha, A., and Nitin, C., 2014. "*Hybrid Approach for Optimizing the Search Engine Result.*", International Journal of Computer Science and Mobile Computing, 3(4):707 – 710.
14. Miller, G. Beckwith, R., Fellbaum, C., Gross, D. and Miller, K. 1990. "*Introduction to wordnet: An on-line lexical database*", International journal of lexicography, 3(4): 235-244
15. Elberrichi, Z. ,Rahmoun, A. and Bentaalah, M. 2008. "*Using WordNet for Text Categorization*", International Arab Journal of Information Technology, 5(1): 16-24.
16. "*Information retrieval*" - Wikipedia, the free encyclopedia, Visited at (Nov. 10, 15).
17. Madhu, G. ,Govardhan, A. and Rajinikanth, T. 2011. "*Intelligent semantic web search engines: a brief survey.*" International journal of Web & Semantic Technology (IJWesT), 2(1).
18. Prakash, K. and Raghavan, S., 2004. "Intelligent Search Engine: Simulation to Implementation." iiWAS.
19. Simon, P. and Sathya, S., 2009. "*Genetic algorithm for information retrieval*", *International Conference of Intelligent Agent & Multi-Agent Systems*, IEEE.
20. Schmitt, L. 2001. "*Theory of genetic algorithms.*" Theoretical Computer Science, Elsevier, 259(1): 1-61.
21. Al-Khateeb, B., Jamal, A., and Jalil., A. 2014, "*Investigating Crossover Techniques for the 8-Queens Problem*", International Journal of Advanced Research in Computer Science, 5(5):118-121.
22. Sastry, K., Goldberg, D. and Kendall, G. 2005. "*Genetic algorithms: In Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*", Search methodologies, Springer US, [pp. 97-125].
23. Mitra, M. and Chaudhuri, B. 2000. "*Information Retrieval from Documents: A Survey*", Information Retrieval, 2:141-163.
24. Guha, S., Rastogi, R., and Shim K. 1999, "*ROCK: A robust clustering algorithm for categorical attributes.*" Data Engineering, 1999. Proceedings., *15th International Conference on*. IEEE.
25. "*Information retrieval*", Oxford Dictionaries- Dictionary, Thesaurus, & Grammar, Visited at (Des. 15, 15).
26. Miller, G. 1995. "*WordNet: A Lexical Database for English*", Communications of the ACM,. 38(11): 39-41.