Effective Search in Online Knowledge Communities: A Genetic Algorithm Approach

Xiaoyu Zhang

Thesis presented to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements of the degree of

Master of Science

in

Computer Science and Applications

Committee:

Weiguo Fan

Gang Wang

Madhav Marathe

September 11th, 2009

Blacksburg, Virginia, USA

Keywords: Online Knowledge Community, Social Network, Genetic Algorithm, Information System, Information Retrieval, Knowledge Adoption Model

Effective Search in Online Knowledge

Communities: A Genetic Algorithm Approach

Xiaoyu Zhang

Abstract

Online Knowledge Communities, also known as online forum, are popular web-based tools that allow members to seek and share knowledge. Documents to answer varieties of questions are associated with the process of knowledge exchange. The social network of members in an Online Knowledge Community is an important factor to improve search precision. However, prior ranking functions don't handle this kind of document with using this information. In this study, we try to resolve the problem of finding authoritative documents for a user query within an Online Knowledge Community. Unlike prior ranking functions which consider either content based feature, hyperlink based feature, or document structure based feature, we explored the Online Knowledge Community social network structure and members social interaction activities to design features that can gauge the two major factors affecting user knowledge adoption decision: argument quality and source credibility. We then design a customized Genetic Algorithm to adjust the weights for new features we proposed. We compared the performance of our ranking strategy with several others baselines on a real world data www.vbcity.com/forums/. The evaluation results demonstrated that our method could improve the user search satisfaction with an obviously percentage. At the end, we concluded that our approach based on knowledge adoption model and Genetic Algorithm is a better ranking strategy in the Online Knowledge Community.

Acknowledgements

This thesis arose in part out of years of research that has been done since 2007, fall. During that time, I have worked with a great number of people whose contribution in assorted ways helped in the research and made me obtain a great research experience.

I would like to express my appreciation and sincere gratitude to everyone who helped me on this thesis. I would like to thanks my advisor, Dr. Weiguo (Patrick) Fan. His serious research attitude, deep understanding of our area and detailed guiding always help me deliver high quality experimental results. His idea always inspired me to finish the current task well, and more important to be a good researcher in the future. He is such a nice and excellent mentor both for research and for life. I would also express my gratefulness to my committee members, Dr. Gang (Alan) Wang, and Dr. Madhav Marathe. Dr. Wang spent a lot of extra time help me resolve the problem I met. His passion and experience to the research helped me improve myself further than I expected. His precious advice to research method and writing will let me benefit for rest of my career. I especially want to acknowledge Dr. Madhav Marathe, who are more like mentors to me, giving me guidance, believing in me even when I didn't believe in myself and always gave me his precious time when I turn to him for extra help in any aspects. His scientific spirit is what I should always learn and respect.

I gratefully thank Virginia Bioinformatics Institute NDSSL Lab supported me the GRA in the past one and half years, which makes me possible to finish my graduate study in Virginia Tech.

I also want to thank my group member Jian Jiao and Xiamo Liu for their kind support and help. Jian spent great effort on help me control the experiment quality. Xiaomo spent lots of time on collaborating with me to remotely label the search results, which I really appreciate. I also quite enjoy the discussion with both of them, which inspired me to come up with a better solution.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 An introduction to the Online Knowledge Community

The Online Knowledge Community (short as *OKC*), also known as virtual community, is defined as an online community that primarily supports knowledge seeking and learning activities based on common interests. It is a web based tool used for a variety of internet users or groups to get together to discuss problems or shared information in virtual environments across time and space. The knowledge base of an OKC is represented by a collection of posts created by its members over a long period of time. The posts are organized into threads, each of which focuses on a particular topic defined by the initial post.

1.1.1 Online Knowledge Community Evolution and Classification

The technologies supporting OKC have evolved from traditional message boards, newsgroups into more feature-rich interactive communication systems. In the early 1990s, the OKC with Web 1.0 tech were coming, such as Theglobe.com (1994), Geocities (1994), and Tripod (1995). These Online Knowledge Communities used tools like chatting rooms and personal information homepage to get people together to interact. The Web 2.0 wave of OKC arrived in the early 2000s and is essentially characterized by virtual communities such as Yahoo! Answer, Google

Group, Flickr, and Facebook. Different OKC have different levels of interaction and participation among their members. The variety of the Web 2.0 virtual communities ranges from adding comments/tags to a blog/post to competing against other people in online video games. Name a few but not limited to: Flicker is a photo sharing community, where users post their own photo shots, comments others' photo to discuss photograph technique; Google Group is a discussion forum. An information seeker will post a question to start a new thread. Other users interested in the topic can reply within the thread. The thread keeps growing as the users dig into the discussion, until they find the solution or give up the discussion. The information seeker picks the best answer from all the replies and labels it. Our study investigated an OKC which is similar as Google Group.

1.1.2 Online Knowledge Community Structure

A large-scale OKC may have several categories, and each category may have several sub-forums, each of which represents a particular topic. For example, the www.vbcity.com/forums/, an online forum mainly focuses on discussing the Visual Basic.NET and Visual Basic 6 developing techniques, have two big categories and eighteen sub forums **Figure 1**. Another example www.gamedev.net/community/forums/ , which is the leading game developer forum, mainly discusses game developing library, computer graph and game design. This online forum has fourteen sub forums as shown in **Figure 1**.

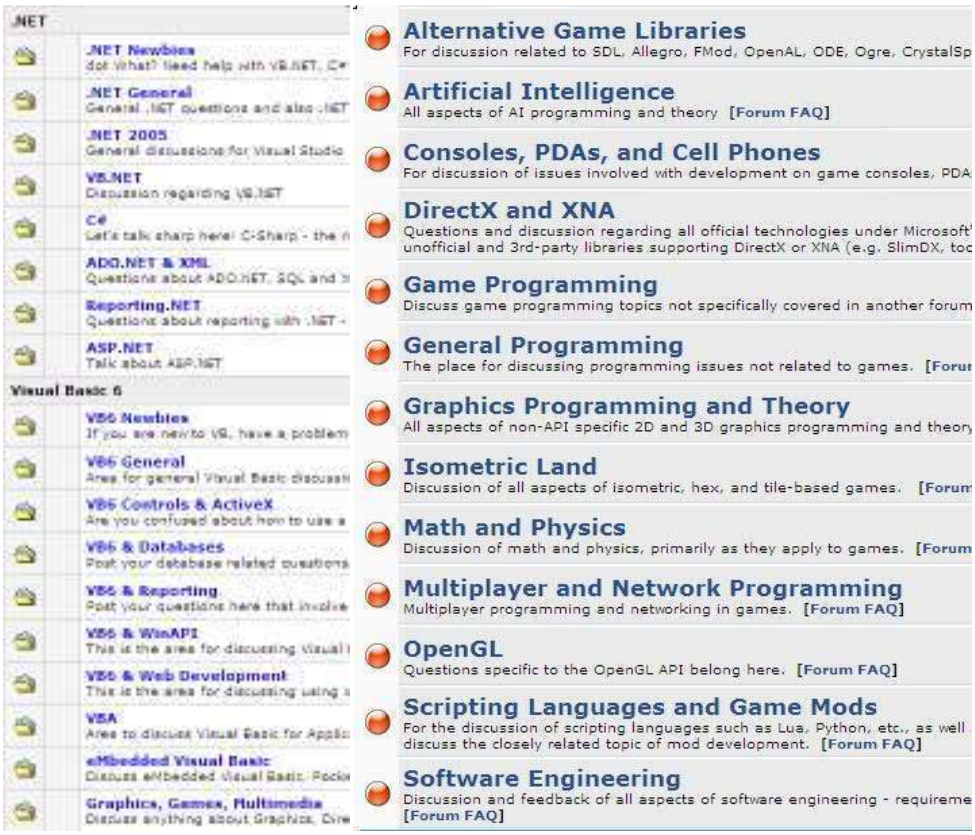


Figure 1 Sub Forums Structure of VBCity and GameDev

1.1.3 Users of the Online Knowledge Community

A long-standing OKC usually has a large member base with a large amount of social interactions made through online discussions over time. For example, www.vbcity.com/forums/ has around 240,546 users and 680,192 posts till July, 2009.

According to members' objectives, participations and contributions in the community, they can be classified into different role. Take VBCity as an example, the users are classified into seven levels of roles from high to low:

- vbCity Leader
- Guru
- Fanatic Member
- HyperActive Member
- Junior Member
- Member
- New Member

Different level of role presents a member's different authority. It has been widely accepted as the hint for readers to judge the authority of the post content, also help user to select the answer among the replies. Although the idea of using this classification is very intuitive, later section will argue that it is far from an accurate way to judge the post content authority.

1.1.4 Online Knowledge Community Analysis Tool:

The purpose of the OKC analysis tool was to help users find newsgroups relevant to their interests; learn about frequent contributing authors, track the popular topics on the Internet. NetScan, launched in September 1999 by Microsoft, is one of the most famous ones. Its data is based on USENET, a worldwide distributed Internet discussion system. NetScan can be used to:

- Find newsgroups where others share your unique interests.
- Monitor the health of newsgroups related to your interests and pursuits.
- Stay informed on current events and the latest trends.
- Locate sources for technical assistance and information.
- Examine troubling issues and hot topics not covered in product documentation.
- Track the participation of your favorite authors across Usenet newsgroups.

1.2 An introduction to the social network:

A **social network** is a graph made of nodes of individuals, which are tied by variety type of interdependency presenting social activities such as friendship, physical exchange, being same location, or partnership. Generally, there are two kinds of ties, directed tie and non-directed tie, so as the types of the network. With the non-directed link, nodes at both sides of the link have equal role; contrarily, with directed link, the relationship carried by the link is only from one side to another side.

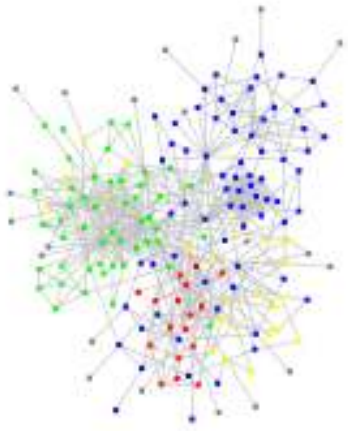


Figure 2 A Small Social Network

1.2.1 Social Network Analysis

Social Network Analysis [SNA] research provides mathematical method to measure the relationship between nodes in the network. To understand a network and its participants, we evaluate the **location** of nodes in the network. Precisely speaking, it is calculating the network centralities of nodes. The network centralities give us the insight to understand questions like "Who are the connectors, mavens, leaders, bridges, or isolates?", "Where are the clusters and who is in them?", "Who is in the core of the network?", and "Who is on the periphery?"

SNA Centralities includes Degree centrality, Betweenness centrality, Closeness centrality, and Clustering Coefficient.

Degree Centrality:

The number of node adjacent (connected by a link) to a given node in a network is the Degree of that node. The normalized Degree centrality is the Degree divided by the maximum possible Degree expressed as a percentage. The equation of Degree centrality [1] is:

$$C_D(n_i) = d(n_i) = x_{i+} = \sum_j x_{ij} = \sum_j x_{ji} \tag{1.1}$$

In the directed network, the Degree centrality is classified as the In-Degree and the Out-Degree. The In-Degree is the number of nodes adjacent to the given node connected by the link pointed from adjacent nodes to the given node. The Out-Degree is the number of nodes adjacent to the given node connected by the link pointed from the given node to the adjacent nodes.

Closeness Centrality:

The Closeness centrality measure is based on the sum of the geodesic distances from each node to all others. The normalized Closeness centrality of a node is the reciprocal of the Farness divided by the minimum possible Farness expressed as a percentage. As an alternative to taking the reciprocal after the summation, the reciprocals can be taken before. In this case the Closeness is the sum of the reciprocated distances so that infinite distances contribute a value of zero. This can also be normalized by dividing by the maximum value. The equation of Closeness centrality [1]is:

$$C_C(n_i) = [\sum_{j=1}^{g} d(n_i, n_j)]^{-1} \tag{1.2}$$

$d(n_i, n_j)$ is the number of lines in the geodesic linking actors *i* and *j*.

$\sum_{j=1}^{g} d(n_i, n_j)$ is the total distance that *i* is from all other actors.

The value of this centrality can reach its maximum at equals $(g-1)^{-1}$, which arises when the node is adjacent to all other nodes; the value can reaches its minimum at 0, which arises whenever one or more actors are not reachable from the actor in question. Since the maximum value of this index depends on *g*; comparison of values across networks of different sizes are difficult. We should make it standardizing as:

$$C_C^{`}(n_i) = \frac{g-1}{\sum_{j=1}^{g} d(n_i, n_j)} = (g-1)C_C(n_i) \tag{1.3}$$

The value ranges between 0 and 1, and can be viewed as the inverse average distance between the node *i* and all the other nodes.

Betweenness Centrality:

Interactions between two nonadjacent nodes might depend on the other actors in the set of nodes, especially the nodes who lie on the paths between the two, which potentially have some control over the interactions between the two nonadjacent nodes. Measure how many geodesics link of pair of nodes the actor lies on.

Let b_{jk} be the proportion of all geodesics linking node *j* and node *k* which pass through node *i*. The Betweenness of vertex *i* is the sum of all b_{jk} where *i*, *j* and k are distinct. Betweenness is therefore a measure of the number of times a node occurs on a geodesic. The normalized Betweenness centrality is the Betweenness divided by the maximum possible Betweenness expressed as a percentage. The equation of Betweenness Centrality [1] is:

$$C_B(n_i) = \sum_{j<k} g_{jk}(n_i) / g_{jk} \tag{1.4}$$

Or

$$C_B^{`}(n_i) = C_B(n_i) / [(g-1)(g-2)/2] \tag{1.5}$$

$g_{jk}(n_i)$ is the number of geodesics linking the two actors that contain actor *i*

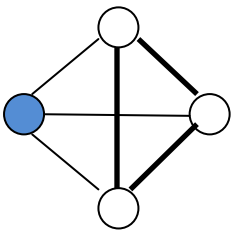
g_{jk} is the number of geodesics linking the two actors.

Clustering Coefficient:

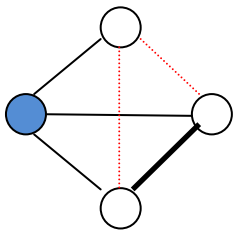
The "overall" graph Clustering Coefficient is simply the average of the densities of the neighborhoods of all of the nodes. The "weighted" version gives weight to the neighborhood densities proportional to their size; that is, nodes with larger neighborhoods get more weight in computing the average density. Since larger graphs are generally (but not necessarily) less dense than smaller ones, the weighted average neighborhood density (or Clustering Coefficient) is usually less than the un-weighted version. Formally, it has been defined as:

$$C = \frac{N_t}{N_{ctv}} = \frac{3 * N_{ct}}{N_{ctv}} \tag{1.6}$$

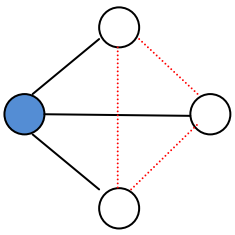
N_t is number of triangles connected to the node, N_{ctv} is number of connected triples vertices. N_{ct} is number of closed triplets. Take an example:



Clustering Coefficient = 1



Clustering Coefficient = 1/3



Clustering Coefficient = 0

1.3 The Social Network inside the Online Knowledge Community

The OKC links people together to exchange knowledge, which form social network. The social network is becoming the important base of OKC.

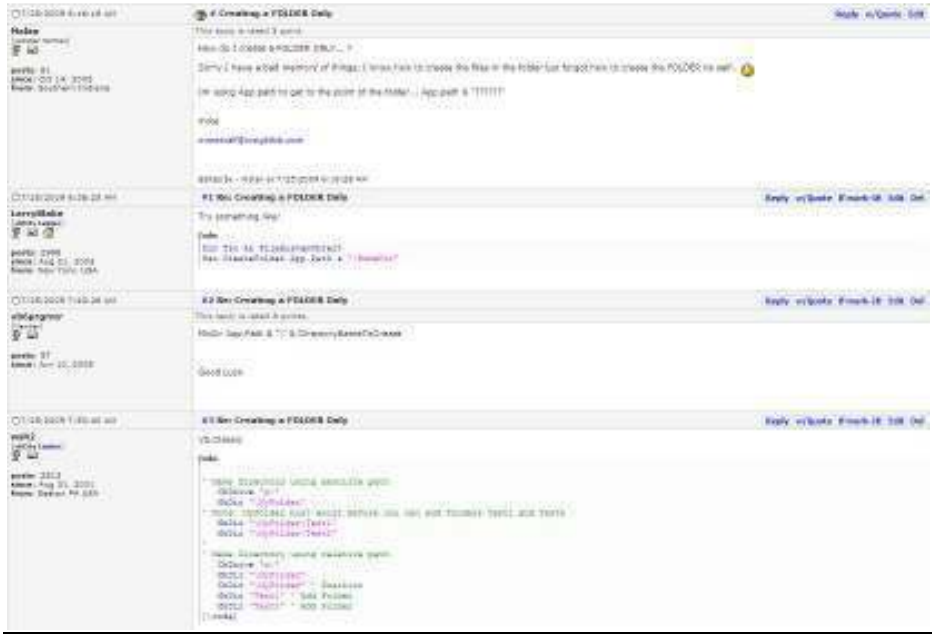


Figure 3 A thread in www.vbcity.com/forums/

There are some rules to follow when we built the social network in our study:

(a) Each member is a node in the network

(b) “Reply to the question” is the only link (relationship) between nodes which arises when a member replies another member in the thread.

(c) We construct the big social network of the entire community by merging social networks of all the threads. There is no concept of thread in the big social network. Only nodes represent the users, and links represent the reply relationship.

(d) Directed and un-weighted social network, which means that multiple links from one node to another node only count one links. However, two nodes point each other count two links.

(e) User replies his own question doesn’t generate any more links.

1.4 The Limitation of Current Ranking Functions in Online Knowledge Community

1.4.1. The Classification of Current Searching Strategies

There are a number of ranking strategies hired by current search engine. They could be classified into three categories:

- Content based ranking strategies

  They use the words lexical statistics data or words syntactical statistics data, like *tf* (token frequency), *df* (document frequency), in a collection of documents to perform the ranking. Including but not limited to, Pivoted TFIDF [2], Okapi BM25 [3].

- Link based ranking strategies

  They use the hyperlink linking the web pages to assign the ranking scores to documents. PageRank[4] and HITS[5] are two well known ones.

- Structure based ranking strategies

  This kind of ranking strategy emphasizes the structure position the words appear in the document. For example, in news search, words appear in the title will be assigned higher weight than others words appearing in the body.

In our study, we define a thread as a single document (a retrieval unit). There are some special features embedded in this kind of documents, which make the above search strategies don't apply to documents in the OKC very well:

1.4.2 There are Few Hyperlinks among the Threads

Different from other Internet web pages, there are few hyperlinks among the threads in an OKC. If we draw a graph by using thread as the nodes, and the hyperlinks as links, most of the nodes in the graph will be isolated.

However, link based ranking algorithms like PageRank explore the intern-relationship of the web pages by its hyperlink structure. They rely on the uniquely democratic nature of the web by using its vast link structure as an indicator of an individual page's value, yet which is missed in the OKC documents. Further, it is also very hard to find the corresponding concepts belonging to the link based algorithm, like the authority page or the hub page, in the OKC documents. Hence, the affects of link based ranking function cannot be promising.

1.4.3 Document Relevance Doesn't Mean Documents Helpfulness

Whether a thread answers the question asked by a query cannot be simply judged by the text relevancy between them. It is highly possible that all the replies describe the question without reaching a solution, though it still makes the high text relevancy. This also matches our experiment experience. During the labeling, we found that in some threads, the starting user could not make the question understood very well. The following repliers asked again or tried to describe the problem from their understanding, which increase the appearance times of the corresponding keywords of the query.

Another reason the text relevancy cannot bring helpfulness is ranking function based on Pivoted TFIDF always prefers the short thread. Even our baseline ranking function, Okapi BM25, also

can be manipulated by the short thread with high *tf*, *df* value. During our experiment, there is one thread, appeared multiple times in the top 50 retrieved list returned by Okapi BM25, but it is never labeled as helpful for any query includes it into search results. However, as our common sense, a short thread normally fails to give a detail solution.

1.4.4 Social Interaction Data Is Not Collaborated With Ranking Functions

The OKC contains rich social interaction data for both users and threads. For example, user's activity time since joined the community, user's total posts number, number of question asked by a use and so on, thread's word count, number of users involved in a thread and so on. These data are not available in other kind of Internet pages, nor do the three ranking functions use them.

However, above features either can be used to identify a thread's content quality or a user's trustfulness, further identify documents' helpfulness. We will discuss the detail in later section of the paper.

1.4.5 Social Network Measures Are Not Collaborated With Ranking Functions

The members social network is the most distinguish difference between the OKC documents and the others types of Internet web pages documents. Due to the social network social capital theory [6], different position of the nodes in the network gives the node different importance in the network. There are some examples of possible important nodes: nodes in the position has shortest path to reach all the other nodes, nodes in the position having the large number of shortest paths passed, nodes in the position only has out-going links. Contrarily, there are examples of possible unimportant nodes: isolated nodes or nodes only having in-going links.

1.4.6 The Search Engine used by VBCity Forums

Here is a screen shot of VBCity Search Engine Advance Search Interface.

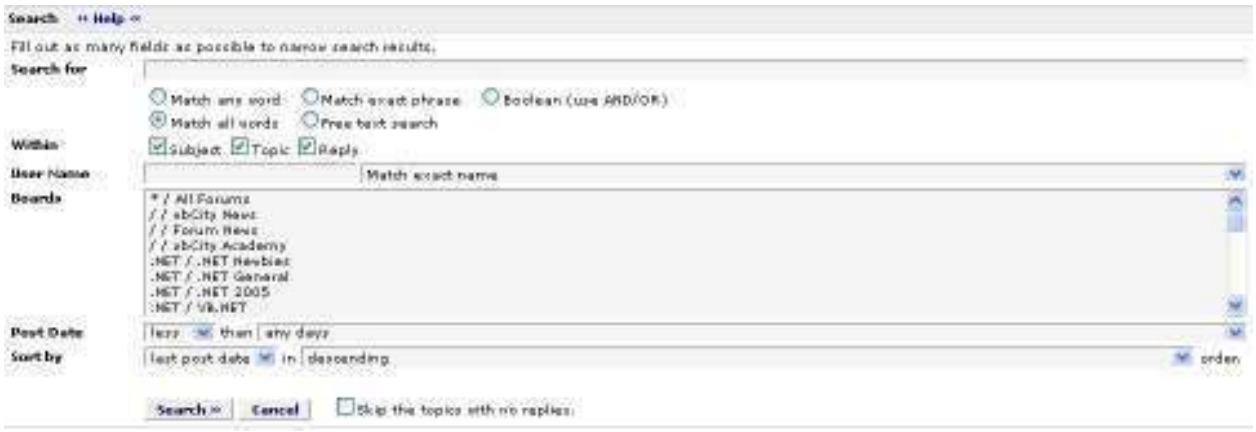


Figure 4 VBCity Search Engine Advance Search Interface

In the advance search interface, besides the keywords search options, a user can also specify search results only include the threads written by a specific user, or the threads belong to one or several specific sub-forums.

However, it is only a content based search engine. The search power is quite limited. Besides the limitation mentioned above, several drawback can be observed:

- This search engine could not understand a question query. We tried several queries we used in the experiment, there is few or even zero relevant results could be found in the retrieved list.
- This search engine defiantly ranks results by the latest date. There is no guarantee of the user's satisfaction by doing this. It is very slightly possible that the latest contributed

threads include the answer to a question a user asks. A user has to filter the searching several times to get a possible result he/she wants.

Several other online forums hire Google as the search engine. After several trails, we could clearly feel that although some of the problems could be avoided, it still cannot leverage its search power to reach a good user satisfaction because it doesn't overcome the limitations we discussed above.

1.5 Research Questions

Based on the previous discussion, we found the limitation of three search strategies (the content based, the link based, and the structure based search strategies) on the OKC documents. Meanwhile, we found OKC is a special kind of web based tool, whose features (social interaction data and social network data) can potentially be used to improve the ranking function performance. Further I propose the following research questions in our study:

(1) What are the additional criteria to rank documents inside the Online Knowledge Community?

(2) Given the large amount of data and those additional criteria within an Online Knowledge Community, can we come up with an optimized ranking strategy to get a better search precision?

Chapter 2

Related Work

In this section, we review the previous research work related to the information retrieval, social network, Online Knowledge Community, and the knowledge adoption model.

2.1 Search in Online Knowledge Communities and discussion forums

From the perspective of information retrieval, the difference between searching World Wide Web by Google and searching Online Knowledge Community is that user could ask his/her specific question directly to someone who already has relative experience or expertise. Hence, locating experts inside Online Knowledge Community is obviously helpful to improve the precision of the search engine when it searches Online Knowledge Communities. A lot of researchers have devoted their effort on developing an expert finder system, which can help others find appropriate expert to answer a question. The previous research works include “Answer Garden” by McDonald and Ackerman [7, 8]; “Contact Finder” by Krulwich B and Burkey C [9]; Krulwich described an approach to manage knowledge by focusing on self-organized activities of the organization's members , which is called “expertise sharing” [10]; Kartz et al. explain the power of expertise locating for information retrieval by stating “Searching for a piece of information becoming a matter of searching social network for an expert on a topic with a chain of personal referrals from the searcher to the expert” [11].

Although above researchers did a good job on exploring the relationship of expertise of social network for the purpose of information retrieval. Either they only focus on finding experts or they didn't combine finding experts with the searching knowledge in Online Knowledge Communities.

Xi et al. tried to use the Linear Regression and Support Vector Machines (SVM) to combine a few thread features and user information to create a ranking function for a newsgroup. However, this work only presents improvement in one of the IR performance measures over the baseline system. Plus, they didn't exam their theory in the context of general OKC.

2.2 Online Knowledge Community Social Network Analysis

Community structure is important property of the social network. In the community structure, nodes are tightly joined together in knit groups (dense connections), between which only existing few nodes connected to the knit groups in a loose manner (sparser connections). Radicchi et al. give a general and quantitative definition of community network structure [12]. From the perspective of network topology, detecting the community network structure will help identify and further analyze the social network. Abramson, G. et al. analyzed a social network come up with plays of a video game [13]. It discovered the different topology between the social network and other networks, from regular lattices to random graphs. Also, this research work revealed the user behavior pattern is governed by the topology of the community. Girvan et al. propose a algorithm to detect the community structure [14]. They tested the network both in real world social network and simulated social network. The results shown that the algorithm could identify the community even the community structure is not well known. Guimera et al. studied a

university's email network evolution process [15]. The results suggested that self-organized social network, like email network could become into a state where the distribution of community sizes is self-similar. Newman et al. tried to come up with a quantitative algorithm to measure and detect the strength of community structure [16, 17]. In the first of their work, he designed an algorithm to divide the network into community structure by remove the edges, which is identified using one of a number of possible "betweenness" measures; in their second work, he expressed the social network measure "modularity" into modularity matrix (set of terms), which is used by his new designed algorithm. The results showed the benefit of the new algorithm which returns higher quality results.

Besides on mining the social network structure, researchers also used social network to examine Online Knowledge Community. Kou and Zhang [18] use social network structure to analyze the asking and replying network structure on a bulletin board system. Chiu et al. used social network as the major tool to analyze a professional virtual community [19]. It integrates the Social Cognitive Theory and the Social Capital Theory to investigate the possibility of members' willingness to share knowledge with others. Y. Li et al. used social network as tool to analyze the community members' knowledge sharing pattern [20].

It is an emerging study in the late 1990s to use statistics method to analyze the Online Knowledge Communities or online social network. The earliest attempt is on Usernet [21], which is a online newsgroup. Sack et al. focused mainly on visualization techniques to understand the social and semantic structure of Usernet [22]. Whittacker et al. [23] conducted an statistical data analysis on the Usernet newsgroups. Findings in their study include unequal levels

of participation in newsgroups and cross-posting behaviors. Later on, Microsoft launched the Netscan website, which is a Usernet groups analysis/mining tool. The functions of Netscan include help users find their interested groups, identify the frequent posting authors, and track the growing of the threads. MA Smith et al. analyzed the tool and used it to mine the Usernet Groups in several dimensions [24-27]. Meanwhile, N. Matsumura et al. and Goh et al. studied others bulletin board systems by statistics method [28, 29]; Maloney-Krichmar used a two and half year statistics date of a online health support community to understand the community's members dynamics interaction and the relationship between online participation and the individual's off-line life [30]. Vicen et al. analyzed a technology news website Slashdot[31]. The Kolmogorov-Smirnov statistics tests were used. Important findings include degree distributions are better explained by log-normal instead of power-law distributions. By presenting the structure of discussion threads with radial tree, they also found that threads show strong heterogeneity and self-similarity.

Different from using statistics method to analyze social network, network centrality is commonly used as the indication of the importance of an individual in the network. In our study, we use the network centrality measures defined by Batagelj [1]. In our study, we use the common network centrality measures: Degree, Betweenness, Closeness and so on. We will explain the definition in the later section of the paper.

Generally speaking, in the context of Online Knowledge Community, members with high in-degree are the knowledge seekers, members with high out-degree are the knowledge contributors who actively response to others' posts. A. Vilpponen et al. found that in electronic community

network, higher in-degree centrality significantly reduces the adoption times [32]. Morrison et al. confirmed that higher out-degree centrality increases the decision time for knowledge adoption [33].

Different from Degree, definition of Betweenness is not so straightforward. Lots of research works are interested in finding the relationship between this centrality measure and information flow path way. Newman found that a social network, a node with high Betweenness will have significant influence over the spreading the information through the network. From another view of information flow, A. Mehra et al. resolved that a node with high Betweenness as one who is able to obtain information from most other members in the network [34]. Based on this, Newman concluded that Betweenness centrality measure can be an indicator of who the most influential people in a social network are [35].

2.3 Knowledge adoption model

The online forums always provide a score to represent the authority or trustfulness of the user. Normally, this score and user ID will be displayed together besides the user's post. The purpose of displaying this score is giving reader a hit of how trustful the post content is coming from. However, we cannot rely on this score since it cannot reflect the user's authority and trustfulness in a right way. Every forum uses their Ad-hoc way to calculate it. For example, in www.vbcity.com/forums/, the question post user can give score to the any of the reply, the corresponding reply user gain this score and use it to calculate his global user rating score.

Users of Online Knowledge Communities often come to seek knowledge that can be applied to their own applications. Therefore, we consider the search behavior in a knowledge-based

community as a knowledge adoption process. A knowledge seeker is interested in not only information relevant to his/her query, but information that he/she is able to comprehend and has confidence in the credibility and authority of the source. Based on the Elaboration Likelihood Model (ELM) [36], Sussman et al. proposed the Knowledge Adoption Model (KAM) [37] **Figure 6**. They found that a person's intention to adopt knowledge was determined by two factors: the argument quality of the received message and the source credibility of the information source. According to the ELM, argument quality is the critical determinant of information influence when the message recipient is able to comprehend the message well. When the recipient is either unable or unwilling to process the message, indications of source credibility become a more critical role in the information influence process. However, the mechanisms underlying the interactions of the two factors in the information influence process appear to be complex [37].

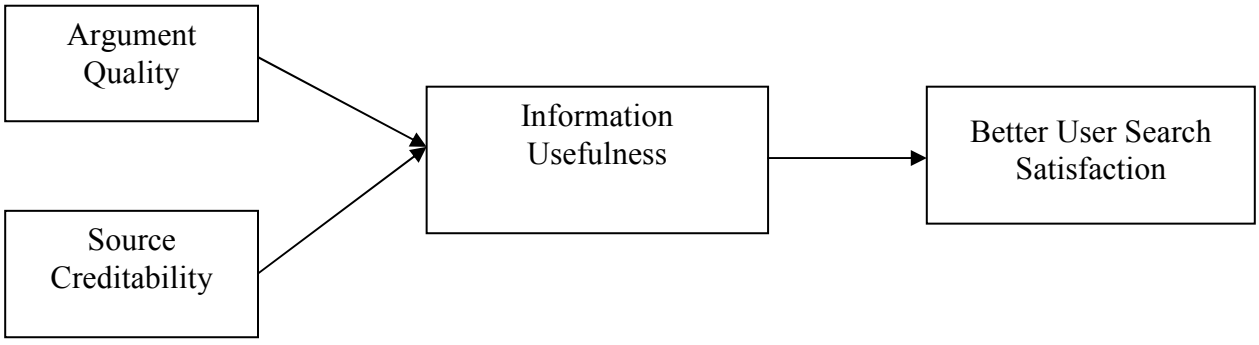


Figure 5 Knowledge Adoption Model

Chapter 3

New GA-based Ranking Optimization Algorithm for Online Knowledge Community Search

We will go through the details of our proposed ranking optimization algorithm. The framework of our system is in **Figure 6**.

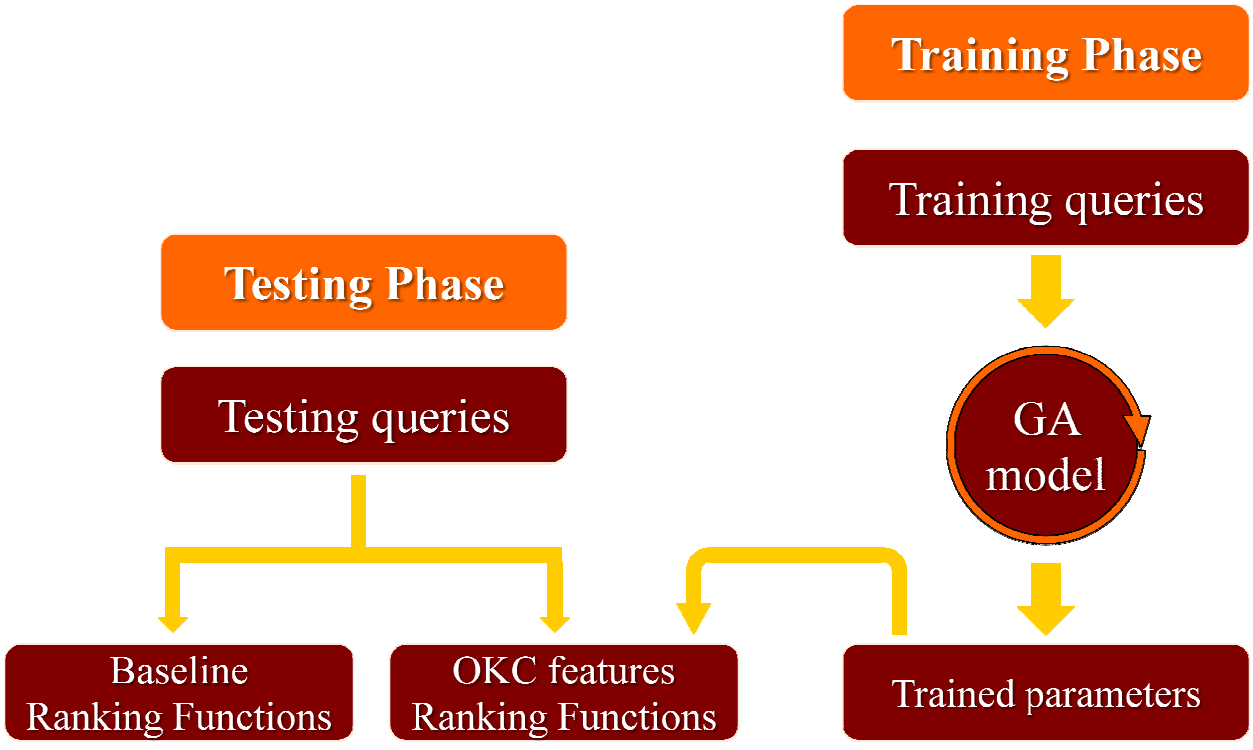


Figure 6 System Framework

3.1 Introduction to the Genetic Algorithm

Genetic Algorithm (GA) [38] is an evolution algorithm inspired by biology evolution process. It is one of the machine learning techniques.

GA is used in computing to approximate the global optimization solution. The basic element in GA is a population of abstract representations (called the chromosomes, the genotype, or the genome). Each individual of the population is a candidate solution. Similar to the biology evolution process, the population in the GA will evolve by producing the next generation. Normally, the first generation will be the randomly generated individuals. In each generation other than the first one, the fitness (quality) of every individual in the population is evaluated by a specific function (called fitness function). Then N best fit individuals will be directly selected into the next generation (called reproduction). Meanwhile, multiple individuals are stochastically selected from the current population, modified (called crossover) to form a new population, and varied (called mutation) from one to another. The new population is then used in the next iteration of the algorithm.

During the evolution, the global best fit individual (chromosome) or several best fit individuals are kept. The algorithm will be terminated when the global optimal solution meets the design value or when the number limitation of generation number is reached, or sometimes when the time limitation is met. To implement a GA, two key requirements are:

1. A fitness function, which is used to evaluate the fitness of the candidate solution. In the later section we will discuss that the fitness function is always problem defined. The design of a fitness function directly impacts its GA optimization performance.
2. An abstract representation model, also known as chromosome. The definition of the chromosome is also depends on the problem. Since each chromosome in the population is a candidate solution, the definition of chromosome should include all the elements that could be used to optimize the problem.

The process of GA can be presented in below pseudo-code:

---

(1) Randomly generate (choose) the initial population

(2) Evaluate the fitness of each individual in the initial population

(3) Repeat until termination by the generation number limitation is reached, time limitation is reached or sufficient fitness is achieved.

(3.1) Select global top fitness score individuals for reproduction (reproduction)

(3.2) Create new generation through crossover and/or mutation (genetic operations)

(3.3) Evaluate the fitness of each new individual

(3.4) Replace the least fitness score individuals with new individuals

---

GA operators

Mutation

In GA, mutation is a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next. It is analogous to biological DNA mutation.

The purpose of mutation in GA is to avoid the algorithm getting trapped in a local optimal solution by preventing the population of chromosomes from becoming too similar to each other. This reasoning also explains the fact that most GA systems avoid only taking the fittest individual of the population in generating the next but rather a random selection with a weighting mechanism toward those that are fitter.

A common method of implementing the mutation operator involves generating a random variable for each bit in a sequence. This random variable tells whether or not a particular bit will be modified from its original state.

Crossover

Crossover is considered to be the Prime distinguished factor of GA from other optimization techniques. Crossover is a genetic operator that combines (mates) a pair of chromosomes (parents) to produce two offspring. It is analogous to the biological crossover.

The idea behind crossover is that the new chromosome may be better than both of the parents if it takes the best characteristics from each of the parents. Crossover occurs during evolution according to a user-definable crossover rate.

A common implementation of crossover operator involves choosing two individuals from the population, and breeding new offspring by combining portions of these individuals. Applying special selection pressure on the parent selection has been recognized as more effective than selecting parents randomly to improve the average problem solving quality of the population [39].

3.2 Prior Research of GA on Information Retrieval

GA has been used in information retrieval optimization since 1990s. Previous research works have applied GA into a wide range of Information Retrieval topics, for adaptation purpose.

Chen et al. applies the GAs in documents indexing [40].

Previous research works also apply the GA in query optimization field of information retrieval. Smith et al. use GA to create better Boolean query [41]. Yang et al. and Horng and Yel et al. introduce a novel approach to automatically retrieval keywords and adapt the keywords weight to improve the search precision [42, 43]. The experiment shown that using GA based approach to tune the retrieval keywords term weight is highly promising for application.

Fan et al. propose a Genetic Programming to tune the term weight strategies for ranking function [44]. In this work, relevance feedbacks were gathered together from user for a specific query to train the Genetic Programming system. Later on the same group further developed a Genetic Programming based optimization of context specific ranking function [45, 46]. The contribution of these research work demonstrate that machine learning tool like Genetic Programming and GA can help us train and discover a better ranking function by tuning the term weight. However, these work hadn't exam the theory in the Online Knowledge Community data set.

Besides trying to optimize the ranking function in the web search design, Lopez-Pujalte et al. proved that retrieval order is a fundamental factor to be considered when design fitness function of the GA, which is used to optimize search by collecting user feedback [47]. Fan et al. exam the possibility to optimize the fitness function of the GA [48]. This work designed four new order-

based fitness functions based on the web search utility theory. And compare these fitness functions with several existing order-based fitness functions on a large web corpus, part of which were shown as very effective.

3.3 Features Used in Online Knowledge Community Search

The chapter 1 has discussed the limitation of three ranking functions on the OKC documents. One of our research questions is "*What are the additional criteria to rank documents inside the Online Knowledge Community?*" We answer it here.

In this section, we propose two types of new criteria based on the Knowledge Adoption Model: Argument Quality and Source Credibility. The objective of these two types of new criteria is to help our new ranking function evaluate the ***helpfulness*** of the retrieved documents.

3.3.1 Argument Quality

The Argument Quality is our concept used to measure the thread content quality like how serious the user's attitude is, the richness of the post text, how many users are contribute to the thread discussion, and how often users contribute to the thread discussion and so on. Based on our observation, we assume that more serious a user's attitude is, or more words in the thread content, or more users contribute to the thread or more often users reply to the thread, more likely that the thread contains the solution to the problem. For the matter of this definition, I extracted below thread social interaction statistics as measures of the Argument Quality:

Dimensions	Features	Descriptions
Relevance	Okapi	Okapi BM25 score
Richness	Length	Number of non-stop words in the thread. Stemming is also used to map related words to the same stem.
	Number of "the", "a"	Number of articles (noun marker)
	Number of replies	Number of posts in the thread
	Number of initial user replies	Number of the initial user reply in his own initialed thread, this could reflect his involvement in the discussion
Completeness	Number of users	Number of distinct users in the thread
	Post per user	Average number of post by every user in the thread
Timeliness	Thread duration	Number of hours elapsed between the first post and the last.
	Posting frequency	Number of posts per hour in the thread
	First response time	Time between initial post and the first reply
	Last response time	Time between initial post and the last reply

Table 1 Argument Quality Features

3.3.2 Source Credibility

The Source Credibility is our concept used to measure how knowledgeable the user is, further how trustful the post content comes from. It is our common sense that if user is knowledgeable, his reply will always be more convincing than others, and the more likely the answer provided by the user is correct. Examples include but not limited to: the more replies the user posted, the more confident the user is sure about his/her knowledge; the more questions the user asked, the more knowledge the user gained in the past; the more frequent the user posted, the more knowledge exchanged between the user and others, which means the user's knowledge gets more frequently updated; the more sub-forums, threads the user participated in, the wider knowledge the user has. Besides those social interaction statistics mentioned above, we also try to use social network measures to evaluate the source credibility (we choose 5 measures, In-Degree, Out-

Degree, Betweenness, Closeness, and Clustering Coefficient). For the matter of this discussion, I extracted below as the measures of the Source Credibility:

Dimensions	Features	Descriptions
Perceived expertise	Activity Length	Total number of days elapsed between first and last posting dates
	Knowledge sharing activity level	Total number of replies made by the user
	Involvement	Total number of threads participated by the user
	Knowledge scope/ broadness	Total number of forums participated by the user
	Number of posts	Total number of posts of the user
	Posting Frequency	Number of posts per year
	Number of questions asked	Number of thread initialed by the user
	Number of answer posts replies	Number of posts in the thread as replier by the user
	Number of questions answered	Number of thread involved as replier by the user
Perceived Trustworthiness	Closeness	A Social Network Analysis (SNA) measure that indicates the communication distance of the member from all other members in the community
	Betweenness	A SNA measure that indicates the importance of the member in forming a coherent communication network
	Out-degree	A SNA measure that indicates the number of people being replied by the member
	In-degree	A SNA measure that indicates the number of people who helped the member (i.e., popularity)
	Clustering coefficient	A SNA measure that indicates how well connected the neighbors of the member are (i.e., brokerage)
	User rating	The average use rating of the member by other members

Table 2 Source Credibility Features

3.4 New Linear Ranking Strategy Discovered by GA

3.4.1 Why do we use GA based approach

The objective of our model is to dynamically utilize the Genetics Algorithm to optimize the features weights, and further create a better ranking function in the context of the OKC. Choosing the GA to build the optimization function is drove by the facts that:

(1) There is no guarantee that assigning the features the equal weights (importance) are the best ranking function. It is intuitive that different feature should have different contribution to the ranking function. Hence, to improve the search performance, we should give the "important" feature higher weight to enlarge its contributions, and give "less important" feature lower weight. However, the 26 features we have are from multiple models and concepts (social network analysis, social interaction data, ranking score of other ranking functions). Hence, it will be extremely complicated for us to figure out the weights distribution. We have to find a way to optimize the weights assignment.

(2) It will be impossible to know the best features weights in advance. Plus, the space of solutions pool (combination of 26 arbitrary real numbers) is infinity. So it cannot be simply solved it by exhausting the entire solution space, which is time consuming and low efficient. However, the nature of the GA helps us explore the solutions pool space in an inductive way, which could generate more diverse and better performance individual in subsequent generations.

3.4.2 The Essential Components in Our GA Based Approach

In order to apply the GA to discover a better ranking strategy, there are several essential components need to be defined. Below lists these key components along with their descriptions:

Ranking Function

A ranking function is the object optimized by our system. We assume the equation of our ranking function as:

$$P = w_1 * f_1 + w_2 * f_2 + ... + w_n * f_n \tag{3.1}$$

P will be the value used to rank documents in our system. The function is a linear combination of all the 26 features. Here $w_1, w_2, ..., w_n$ are the weights assigned to features, which should be non-zero real numbers. The features $f_1, f_2, ..., f_n$ will be the input to the system.

Populations

Each chromosome in the population is a series of 26 non-zero real number, which represents the corresponding weight associated to a specific feature. We set up the range of weights to be from 0 to 100. Our GA creates the initial population of the chromosomes randomly. In each step of the evolution, the number of chromosome in the population is kept the same; the number of genes (weights) of the chromosome is also kept fixed for the whole population.

Fitness function

It is the objective function GP aims to optimize. We will discuss the design of fitness function in the Section 3.5. Basically, we will use six fitness functions and compare their impacts.

Genetic operator

The crossover operator we used is the basic one provided by the API, which randomly chooses two chromosomes (a vector of 26 non-zero real numbers) from the population and "mates" them by randomly picking one gene (a non-zero real number) and then swapping that gene and all subsequent genes between the two chromosomes. The two modified chromosomes are added to the candidate chromosomes.

The mutation operator we used is the basic one provided by the API, which goes through all the genes in the every chromosome in the population and randomly choose two chromosomes to mutate them by the mutation rate provided. The mutated chromosomes are added to the chromosomes.

The reproduction operator we used is a natural selector which simply chooses certain percentage of the best fit chromosomes according to the reproduction rate, and put them into the next generation of population.

Control parameters

The Crossover rate, mutation rate and the population size are important control parameters of a GA system that need to be tweaked well to get a good performing GA. By doing multiple times of preliminary exploratory, we select the optimal combination of control parameters and fix all of them experimentally.

An enough crossover rate is needed since a low rate will not be able to give enough chance to generate new offspring who might be better than both of the parents. However, we could not give too high crossover rate since we choose the characters (genes) from two "mated" chromosomes randomly. There is no guarantee that all the offspring is better than the parents. Hence, too high crossover rate will eliminate the chance to get good chromosomes by other genetic operators. After trying 0.05, 0.1, 0.2, 0.4, 0.6 and 0.8, we choose crossover rate as 0.8 finally.

A reasonable mutation rate is required since a too low rate will not be able to bring enough chance of population variation. However, a high mutation rate will lead unstable performance of evolution since there will be too much unpredicted chromosome coming in each new generation. After trying 0.01, 0.05, 0.1, 0.2, 0.4, 0.6, and 0.8, we choose mutation rate as 0.2 finally.

A good reproduction rate is needed since it will always be a good practice to keep the best N chromosomes to the next generation. If the reproduced population are too less, we may lose some very good candidates in each generation, but if the reproduced population are too many, we will lose the chance of evaluating part of good new generated individuals. Considered the large population we have and after several trails, we finally choose reproduction rate as 0.05.

The large enough population number is required to avoid the early convergence of the algorithm, which means that the algorithm reaches a local optimal solution, not a global optimal solution in the solution space. However, larger population than required will not be able to improve the experiment further and waste the resource. After trying with 500, 1000, 1200, 1400, 1600, and 2000, we choose 1400 population finally.

Stopping criterion

There are two typical stopping criterions: one is stopping the evolution loop when the best individual reaches the designed optimized problem solution, another one is stopping the evolution loop when the number of generation reaches the limitation. Since it is not easy to define the exact optimized problem solution for our problem, we choose the later stopping criterion, which is stopping the evolution loop at the generation 100.

3.4.3 GA Optimization Procedure

With the above setting, we propose our GA based optimization process:

1. Generate the initial population randomly
2. Evaluate the fitness of each individual in the initial population
3. Training Phase: repeatedly perform the following operations until reach 100^{th} generation
   3.1 record the top N best fit chromosomes with their fitness score (reproduction)
   3.2 generate new individuals with genetic operators
      --crossover
      --mutation
   3.3 Evaluate the fitness of each new individual
   3.4 Replace least fitness score individuals with new individuals
4. Testing Phase: Apply the best chromosome genes to the testing data and calculate the search results measures, and compare them with baselines

Figure 7 the GA Optimization Procedure

3.5 Design the Fitness Function

3.5.1 The Importance of Fitness Function for GA

A fitness function is the objective function that quantifies the optimality of a chromosome. It helps the GA obtain the best solution within a large search space. An ideal fitness function correlates closely with the algorithm's goal, yet may be computed quickly. A good fitness function could help GA search the solution space more efficiently. A bad fitness function, on the other hand, can make GA trapped to a local optimal solution. Design of fitness function is subtle in the most cases. In some cases, it is even very hard to come up with a working fitness function.

As we mentioned earlier, the GA has been applied to the information retrieval area with promising results. In our study, we will try to design better fitness functions than the 3 performance measure functions used in the testing phase, by borrowing the theory of order-based ranking measure and utility theory from [48].

The performance measure functions used in the information retrieval can be classified into two types, *top N function*, and *order-based function* [48]. A *top N function* counts only the number of relevant documents in the top N retrieved list. Meanwhile, an *order-based function* not only counts the number of relevant documents, but also the relative position in the top N retrieved list. Prior researches have confirmed that the *order-based function* gets more favorable results[48]. We will try to apply the same theory into designing a fitness function.

Performance measure functions like MAP, MRR, and NDCG are *order-based function*. We will use those functions as the GA fitness functions in the experiment and compare their impacts with the impacts of the utility functions we designed below.

3.5.2 Using Utility Theory to design Fitness Function

The utility theory in [48] states that there exists a user' preference function that assigns a user's preference value for each item. Assuming that for a list of relevant documents, the larger the rank is, the lower the user's preference value it has. The user's preference function is a utility function and the user's preference value is a utility score. We could put it into the formula that given a utility function $U(x)$, and two ranks $x1, x2$, with $x1 < x2$, according to the above assumption, the equation $U(x_1) > U(x_2)$is hold.

/How to design a suitable utility function for the GA system now becomes our major concern. There are lots of possible candidates to satisfy the equation, for example, $f(x) = \log_{10}(1/x)$, and $f(x) = p^x, p \in (0,1)$. However, we could not simply adopt those functions because their values drop too fast while the x increases the value, which cannot make the function be selective to the top ranked documents. The ideal utility function should favor the top-ranked document (keep high value within the small x value range), and then lose its value quickly. Further, the value of the function should be bounded within a certain rang.

3.5.3 The Utility Fitness Function Definition and Equation

Now we go through the definitions of the utility functions we designed. Fan et al. has defined four utility functions for their task [48]. However, the experiment conditions between their research work and our study is different: Fan et al focus on the TREC data set and ranked the top 1000 documents, while our study focused on the OKC documents and only ranked the top 50 documents. The fitness functions has to be adjusted since our experiment has the different favorite top, value dropping range and value bounding range. Three equations of our utility functions are presented as the following:

FFP1

$$Fitness_{FFP1} = \sum_{i=1}^{|D|} r(d_i) \times k_1 \times \ln^{-2.5}(i + k_2) \tag{3.2}$$

$r(d) \in (0,1)$ is the relevance score associated with the document. Its value is 1 if the document is judged as helpful; the value is 0 if the document is not judged as helpful. $|D|$ is the total number of documents retrieved, which is 50 in our study. The exploratory analysis shows that the $k_1 = 5, k_2 = 1$ best fits the design.

FFP2

$$Fitness_{FFP2} = \sum_{i=1}^{|D|} r(d_i) \times k_3 \times \log_{10}(50 / i) \tag{3.3}$$

$r(d)$ and $|D|$ has the same definition as FFP1. I set $k_3 = 5$ in our experiment.

FFP4

$$Fitness_{FFP4} = \sum_{i=1}^{|D|} r(d_i) \times k_4 \times k_5^i \tag{3.4}$$

$r(d)$ and $|D|$ has the same definition as FFP1. I set $k_4 = 14, k_5 = 0.8$

The curves of these three utility functions are shown in the below graph. They all have the similar shapes; however some of them are steeper. The impact of different curve shape on the performance will be analyzed in the further section.

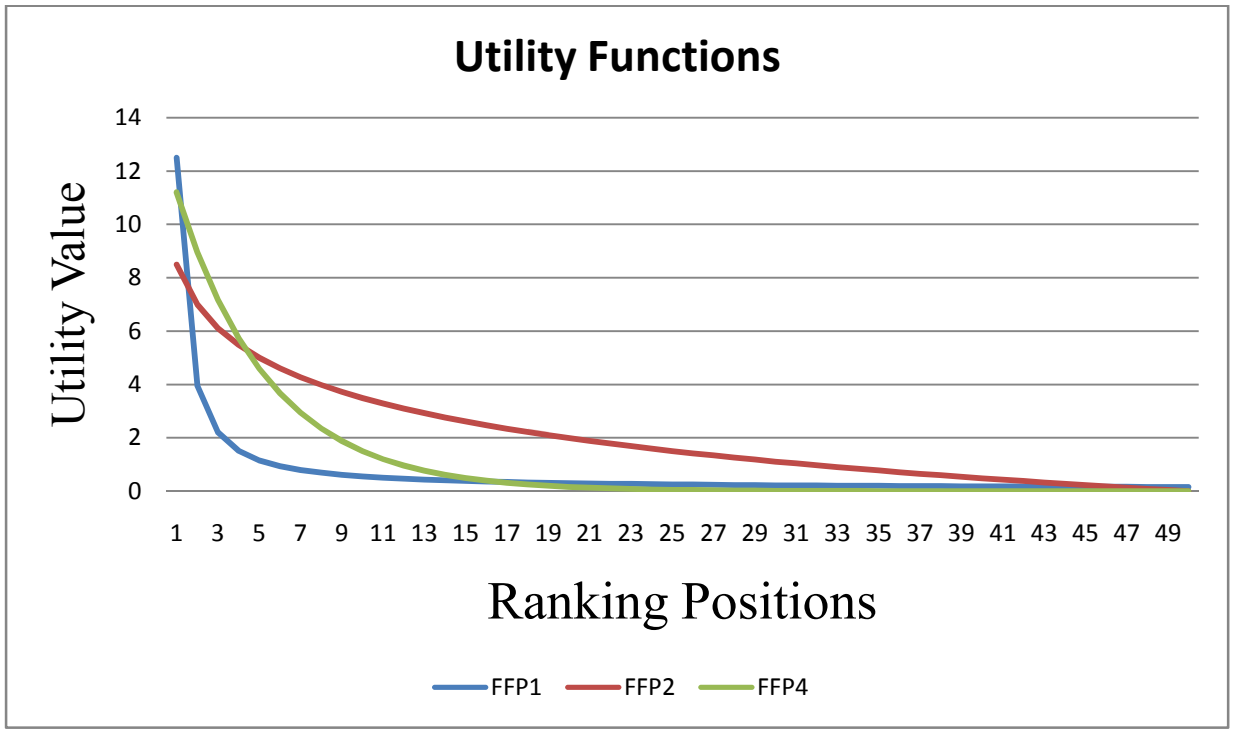


Figure 8 The Utility Function Curve

Let's look at an example of how to use the utility function to evaluate a ranked list.

Given a retrieved rank list 1 (top 10) *1 1 0 0 0 0 0 0 0 1*, the value calculated by FFP2 is:

$$5*\log_{10}(50/1)+5*\log_{10}(50/2)+5*\log_{10}(50/10)=18.9794$$

Given another retrieved rank list 2 (top 10) *0 0 0 0 0 0 0 1 1 1*, the value calculated by FFP2 is:

$$5*\log_{10}(50/8)+5*\log_{10}(50/9)+5*\log_{10}(50/10)=11.1978$$

Chapter 4

Experiments

4.1 Experiment Design

The objective of this experiment is trying to prove that using our GA ranking strategy to search an OKC documents collection can give users better search experiences than the chosen baseline ranking strategies. The Experiment process flow is presented in **Figure 9.**

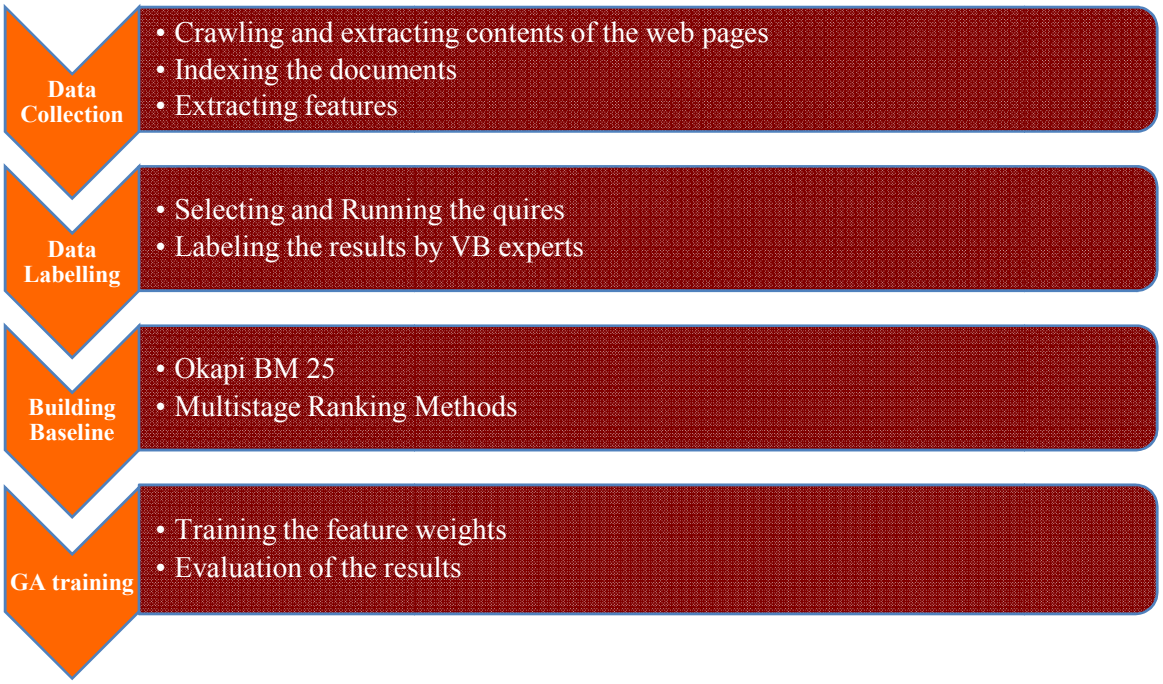


Figure 9 Experiment Process Flow

4.2 Data Collection

Data Crawling, Extraction and Indexing

We choose http://www.vbcity.com/forums/ as our data source. The latest data shows that the forum includes 238479 users, 678426 posts. It is one of the largest online forums discussing Microsoft Visual Basic relative coding questions.

Crawling the Data

We use the Wget tool (http://www.gnu.org/software/wget/) to crawl the web pages to the local disk. The forum http://www.vbcity.com/forums/ only keeps the latest 2000 days threads web pages available online. Since the VBCity forum started around 1999, and the Wget crawled data around Sep, 2007, we roughly have two thirds of the total threads web pages (2002 to 2007).

Parsing the Data

Figure 10 A sample post

After the crawling, we parsed the web pages and put the post text into the corresponding thread profile. A thread profile contains all the posts' text contents. During the parsing, we only kept the text but deleted the quote block (**Figure 11**), the code block (**Figure 12**) and the signature block (**Figure 13**)

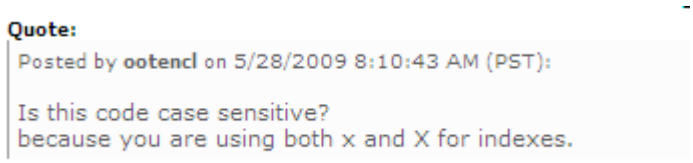


Figure 11 A sample quote block inside the post

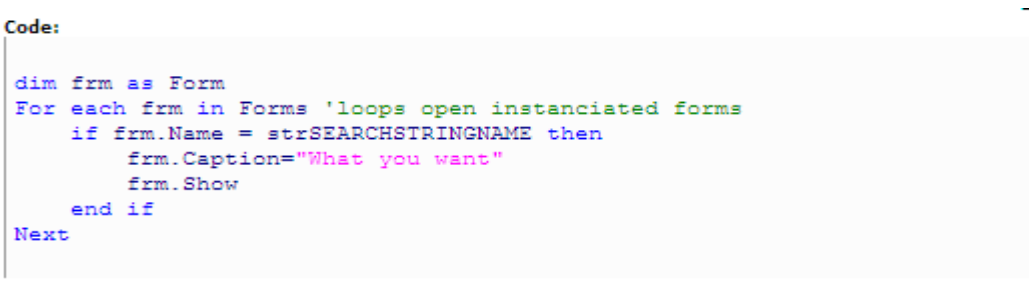


Figure 12 A sample code block inside the post

Figure 13 A sample user signature block inside the post

Indexing the Data

I use the open source Apache Lucene (http://lucene.apache.org/java/docs/) to index the threads profiles. Each thread is an index unit. Each index unit indexes the entire thread profile once and the thread title three times. The reason indexing thread title is because the thread title can reflect the discussion content to some extent. We can simply treat it as the short description of the thread starting post (the post asks the question). The thread title must contain some keywords match the query. We index the thread title three times to increase the chance of match, since the thread title is too short compared with the thread content.

We also indexes thread word count for Okapi algorithm needs.

4.3 Building the Baseline

4.3.1 Okapi BM25

We used the Okapi BM25 ranking function as the first baseline. The Okapi BM25 is a ranking function used by search engine to rank documents based on their relevance to a given query. The Okapi BM25 represented the state-of-the-art retrieval function used in the information retrieval.

The Okapi BM25 ranking function ignores the intern-relationship between the query terms within one document. It ranks documents based on two parameters (1) the query terms appearing in the each document (term frequency), and (2) document length.

The equation of Okapi BM25 could be expressed as:

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \bullet \frac{f(q_i, D) \bullet (k_1 + 1)}{f(q_i, D) + k_1 \bullet (1 - b + b \bullet \frac{|D|}{avgdl})} \tag{4.1}$$

Where $f(q_i, D)$ is the term frequency in the document D, $|D|$ is the document D length (words number), *avgdl* is the average documents length. k_1 and b are free parameters.

$$IDF(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \tag{4.2}$$

Where *N* is the number of documents in the collection, $n(q_i)$ is the number of documents contain term q_i

The above formula for IDF shows potentially major drawbacks when using it for terms appearing in more than half of the corpus documents. Several experimental experiences shown that the long query always bring worse search precision than the short query, which proves this point. The reason leading this behavior is because these kinds of terms will bring negative scores. For example, for any two almost-identical documents, one which contains the term and one which does not contain it, the latter one will possibly get a larger score. This means that terms appearing in more than half of the corpus will provide negative contributions to the final document score.

Multistage Ranking Methods

We borrowed a multistage ranking strategy similar to the one used in [49]. Under this strategy, the query-document relevance is only used to obtain an initial results set of top m documents ranked by the relevance score (in our study, it is Okapi BM25 score), which is referred to as stage 1 results. Other non-relevance-related features like AQ, SC, and AQSC will be used to re-rank the documents in the initial set. As it is unknown how AQ and SC interact for our OKC corpus, we propose four alternative re-ranking methods.

4.3.2 Two Stage Ranking — AQ only

After using Okapi BM25 to return the top 50 in the stage 1, we use the sum of AQ features to re-rank the top 50 documents in the stage 2. The score function that we use to re-rank the m documents in the initial set in stage 2 is:

$$f(AQ\_only) = \sum_{i=1}^{x} AQ_i \tag{4.3}$$

Where AQ_i is normalized between 0 and 1.

4.3.3 Two Stage Ranking — SC only

After using Okapi BM25 to return the top 50 in the stage 1, we use the sum of SC features to re-rank the top 50 documents in the stage 2. The score function that we use to re-rank the m documents in the initial set in stage 2 is:

$$f(SC\_only) = \sum_{j}^{y} SC_j \tag{4.4}$$

Where SC_j are normalized between 0 and 1.

4.3.4 Two Stage Ranking — AQ SC equal

After using Okapi BM25 to return the top 50 in the stage 1, we use the sum of AQ features and SC features to re-rank the top 50 documents in the initial set in stage 2. The score function that we use to re-rank the m documents in the initial set in stage 2 is:

$$f(AQ, SC\_equal) = \sum_{i=1}^{x} AQ_i / x + \sum_{j=1}^{y} SC_j / y \tag{4.5}$$

Where AQ_i and SC_j are normalized between 0 and 1. x is the number of AQ features, y is the number of SC features. The reason we divide features value by features number is trying to give each feature the same weight.

4.3.5 Three Stage Ranking — SC+AQ

The three stage ranking is based on the assumption that non-relevance AQ and SC features may have different importance when contributing to the final document ranking. Research suggests that source credibility has little impact on attitude change towards a message when the message recipient is familiar with the topic [50]. That way the recipient will focus on analyzing the message argument rather than evaluating the expertness and trustworthiness of the information source. Therefore, we argue that SC features may be even noisier than non-relevance-related AQ features. Thus we propose to first re-rank the 50 documents in the initial set from stage 1 using non-relevance-related AQ features only (stage 2). SC features will then be used to re-rank the top

n documents with high AQ ratings (stage 3). The ranking functions for the stage 2 and the stage 3 are listed as the following:

$$f(SC) = \frac{1}{y}\sum_{j=1}^{50} SC_j \tag{4.6}$$

$$f(AQ) = \frac{1}{x}\sum_{i=1}^{n} AQ_i \tag{4.7}$$

4.4 Query Selection

We generated queries by below steps: Firstly we crawled the most recent 800 threads from the Microsoft MSDN Visual Basic, which is also one of the largest Visual Basic discussion forums. Secondly, we parsed the metadata of each thread, which includes the number of replies, the number of views, titles and thread URLs. Thirdly we sorted the threads in the descending order of number of replies. And then picked out those with replies number larger than 5. Fourthly, we manually went through all those threads. And then we filtered those questions which are too specific or too general.

For example:

a. Too specific: "*How to fix a certain error in the code*?"
b. Too general: "*Tutorial about windows application development", "database system development*"

Finally, for each selected thread, rephrased the question post into a question sentence (normally it is a How question).

For example:

> Question Post:
>
> *I'm reading a folder to assess the width x height of jpeg's, to classify them in various ways. Opening each as a bitmap and reading the bitmap width and height works but of course is laborious and very slow ....*
>
> *I've read various JPEG file format papers but retrieving the image pixel width and height seems none too obvious.*
>
> *Is there a logical way in VB to get this info quickly? My jpeg's are bog-standard ones made using VB's "Bmap.Save(Fname, ImageFormat.Jpeg)".*
>
> Query:
>
> *How can I read a JPEG file for image dimensions?*

The reason I filtered the too specific questions is because this kind of question cannot retrieval enough relevant threads in the retrieved results, which may bring too much randomness into the experiment; the reason I filtered the too general question is because this kind of question will retrieval too many relevant threads in the retrieved results. Since high percentage relevant results means the experiment measures' value will be very high, this leads the very limited space to improve the baseline algorithms.

The way I generated the queries bringing the randomness into it, which avoids the bias in the experiment. Also it can make sure there are enough percentage relevant threads in the search result, but not too high.

We finally chose 43 queries in total.

4.5 Label the Search Results

We use the generated queries to against the index files. For each query, return top 50 search results by the Okapi BM25. Each result is a thread in the www.vbcity.com/forums/. The objective of labeling the search results is to generate a "golden standard", which will be used to evaluate the performance of varieties of ranking strategies.

As we discussed above, in the context of the OKC, content relevance cannot make sure the content help the search engine users resolve the problem. Hence, we use "helpfulness" instead of "relevance" to judge the label of the search results of a specific query. Here are the criteria:

(1) Label as 2, **Helpful**, if the thread content discusses the problem asked by the query, and keep the right direction of discussion, and finally provides the solution to that problem.

(2) Label as 1, **Relevant,** if the thread content discuss the problem asked by the query, and keep the right direction of discussion, but failed to provide the solution to that problem.

(3) Label as 0, **Non-Relevant,** if the thread content doesn't discuss the problem asked by the query, or doesn't keep the right direction of discussion the problem asked by the query.

4.6 GA Experiment Procedure

We implemented our GA system by JGAP (http://jgap.sourceforge.net/), which is a well known Genetic Algorithm and Genetic Programming open source Java API. This API has been used in many research works and has gotten good results.

To eliminate the randomness in the experiment, we run the same experiment on three data sets by shuffling the same 43 queries three times. We reported the result for each data set and the average result for all three data sets.

For the same experiment setting on the same data set, we run it five times by giving the GA different random starting seed. The results reported the best one of the five runs. Although the results show that the standard deviation of all the five runs is very small.

The GA tuning process is conducted by two phases: *training phase* and *testing phase*. We have 43 queries in total. We choose 30 queries (about 70%) as training data, and rest 13 queries (about 30%) as testing data. In the *training phase*, the training data is loaded to train the weights associated with the features. In the *testing phase*, weights of the best fit chromosome will be used in our linear ranking function on the testing data to get performance measure value, which will be compared with the baseline values.

Training phase: Six fitness functions (FFP1, FFP2, FFP4, MAP, MRR, and NDCG) were used in the separated training process. The training improvement results and testing improvement results will be used to evaluate the reliabilities of the six fitness functions.

By doing the preliminary exploratory, the best experimental control parameters we chose were: population size as 1400, crossover rate as 0.8, mutation rate as 0.2, and reproduction rate as 0.05. For each run of the GA, we terminated the evolution loop at the 100^{th} generation. The experiment section will tell that the 100 generations of evolution is fairly enough since all the fitness functions value is convergent before reaching 100^{th} generation.

A chromosome is a series of 26 real numbers. For all the 26 features we have, we set their weights range from 0 to 100. At the first generation, random weights (within 0 to 100) are assigned to each of the feature for all of the chromosomes in the population. For each generation, the same number of new populations will be generated by applying three GA operations (mutation, crossover, reproduction).

For each generation, a global best fit chromosome with the highest fitness score selected from this and all the previous generations will be kept as the "Best so far" individual of this generation.

For each generation, we applied three performance measure functions (MAP, MRR, and NDCG) on the "Best so far" individual. We used the sum of three performance measure values (called the SUM_Train) as the judgment criteria to justify the training improvement process on the training data. After last route of evolution, the "Best-trained chromosome" (chromosome with the highest SUM_Train value selected from all the 100 generations) will be considered as the best solution provided by our system.

Testing Phase: The testing data set is used in the phase. We applied the weights of the "Best-trained chromosome" in the linear ranking function on the testing data. The performance of this

linear ranking function will be measured by MAP, P, MRR, and NDCG as the final testing results gained by our GA based ranking strategy.

At the end of the testing phase, final testing results will be compared with the 5 baseline results ("Okapi BM25", "AQ only", "SC only", "AQ SC equal weight", and "Best of SC+AQ").

4.7 Search Result Evaluation

To measure the ranking results, we choose 4 well known measures to evaluate the ranking. These 4 measures are wildly used in information retrieval research: MAP, P, MRR, and NDCG. They measure the different aspects of the ranking. The definitions of these 4 measures are below:

***P* (Precision)**

Given a rank-ordered vector V of results $< V_1,...,V_n >$ to query q, let $rel(V_i)$ be 1 iff V_i is relevant to q and 0 otherwise. The precision of V at document cut-off value k is the number of relevant documents in the top k results:

$$P@k(V) = \frac{1}{k}\sum_{i=1}^{k} rel(v_i) \tag{4.8}$$

This is Top-N based performance measure function.

MAP

Given a rank-ordered vector V of results $< V_1,...,V_n >$ to query q, the average precision of V at document cut-off value k is the mean of the precisions at every relevant document (or 0 if there are none):

$$AP@k(V) = \underset{v_i : i \le k \wedge rel(v_i)=1}{avg} P@i(V) = \frac{\sum_{i=1}^{k} P@i(V) rel(v_i)}{\sum_{i=1}^{k} rel(v_i)} \tag{4.9}$$

The mean average precision of the test set is the mean of the AP's of the queries in the test set. This is an order-based performance measure function.

MRR

Given a rank-ordered vector V of results $< V_1,...,V_n >$ to query q, the reciprocal rank of V at document cutoff value k is:

$$RR@k(V) = \begin{cases} \frac{1}{i} & if \exists i < k : rel(v_i) = 1 \wedge \forall j < i : rel(v_i) = 0 \\ 0 & othrewise \end{cases} \tag{4.10}$$

The mean reciprocal rank of the test set is the mean of the RR's of the queries in the test set. This is an order-based performance measure function.

NDCG

Given a rank-ordered vector V of results $< V_1,...,V_n >$ to query q, let label(v_i) be the judgment of vi (0=worst,2=best)(the reference use 0=worst, 5=best, but due to the label limitation, we use smaller scale). The discounted cumulative gain of V at document cut-off value k is:

$$DCG@k = \sum_{i=1}^{k} \frac{1}{\log_2(1+i)} (2^{label(v_i)} - 1) \tag{4.11}$$

The normalized DCG of V is the DCG of V divided by the DCG of the "ideal" (DCG-maximizing) permutation of V (or 1 if the ideal DCG is 0). The NDCG of the test set is the mean of the NDCG's of the queries in the test set. This is an order-based performance measure function.

Chapter 5

Results and Conclusion

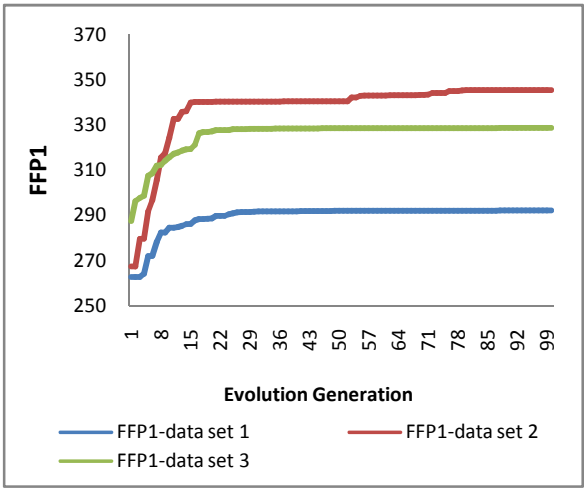
5.1 Training performance improvement

In this section, we report the data of the GA training phase from a number of dynamical dimensions. The **Figure 14** shows the growing curve of six fitness functions we used. These figures describe the range of generations in which the corresponding fitness function grows fastest, and the range of generations in which the corresponding fitness function is convergent. These figures could give us a straight insight on the GA evolution process for different fitness function. The **Figure 14** also helps us compare the evolution process by the same fitness function on different data set.

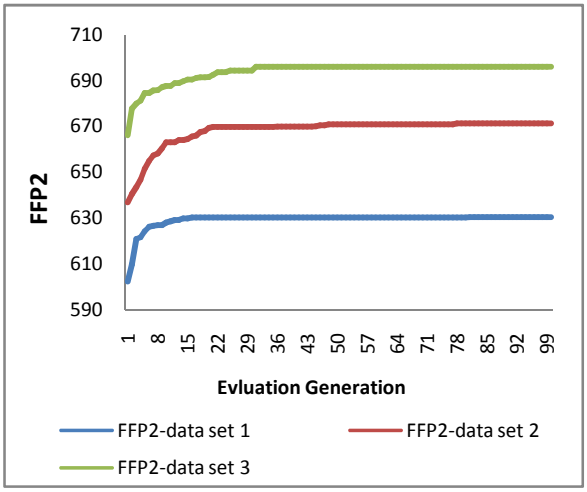
The **Figure 15 ~ 17** measured the training data along with the GA evolution process. Each one of them measures the training data by one of the performance measures (MAP_50, MRR_50, or NDCG_50). Here we draw the separated chart for different fitness function by the same performance measure on three data sets. Comparing the training performance on different data set can help us find out the different impact brought by the three data sets. We can use this information to analyze the GA improvement in the testing phase.

The reason we didn't use P_50 (precision of top 50 results) to measure the training data is because P is not an order-based performance measure function. Hence there is no difference on P_50 value no matter how rank is changed on the training data. In the early section, we also mentioned that for the same matter of fact, P is not qualified to be a fitness function neither.

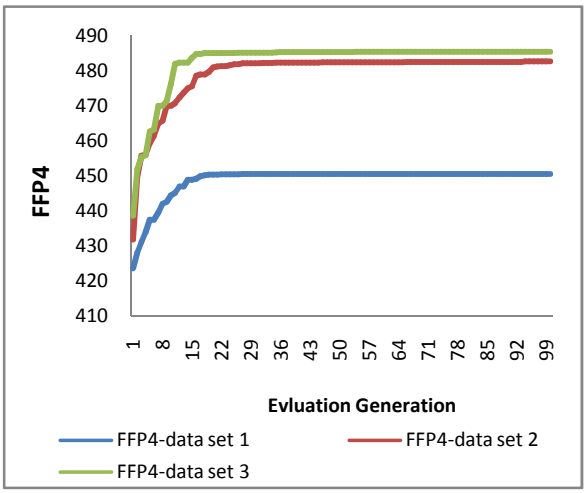
The **Figure 18~ 20** display the same content as the **Figure 15~17** from a different dimension. Each figure of **Figure 18~20** has three charts, each of which illustrates the different impact of six fitness functions for the same performance measure on one of the data sets. The **Figure 18~20** provides the direct evidence to compare the different impact of six fitness functions on the GA training phase.



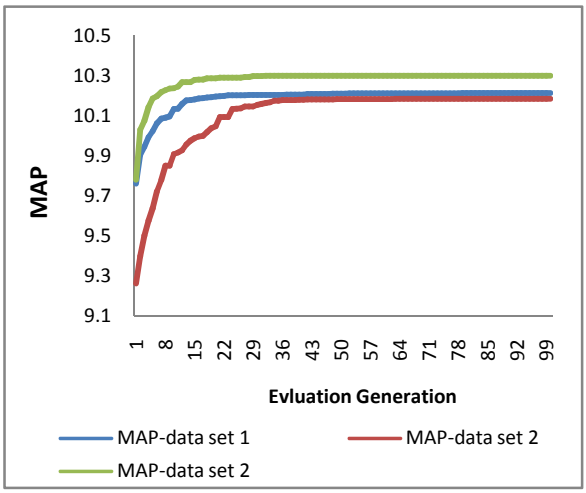
(a)



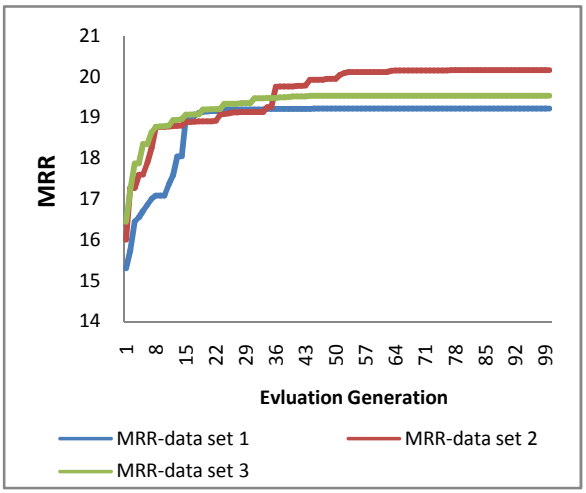
(b)



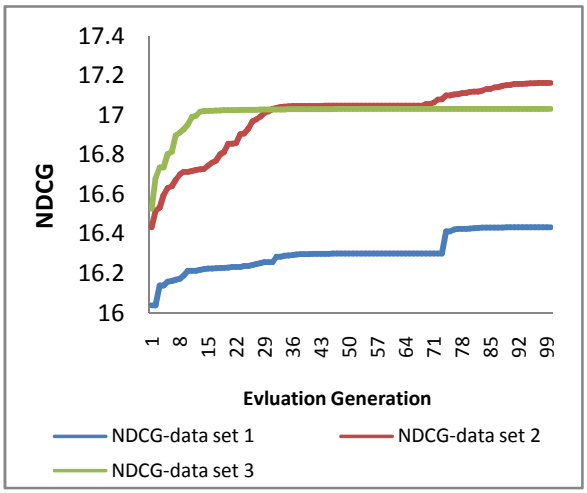
(c)



(d)



(e)



(f)

Figure 14 Six Fitness Functions growing curve

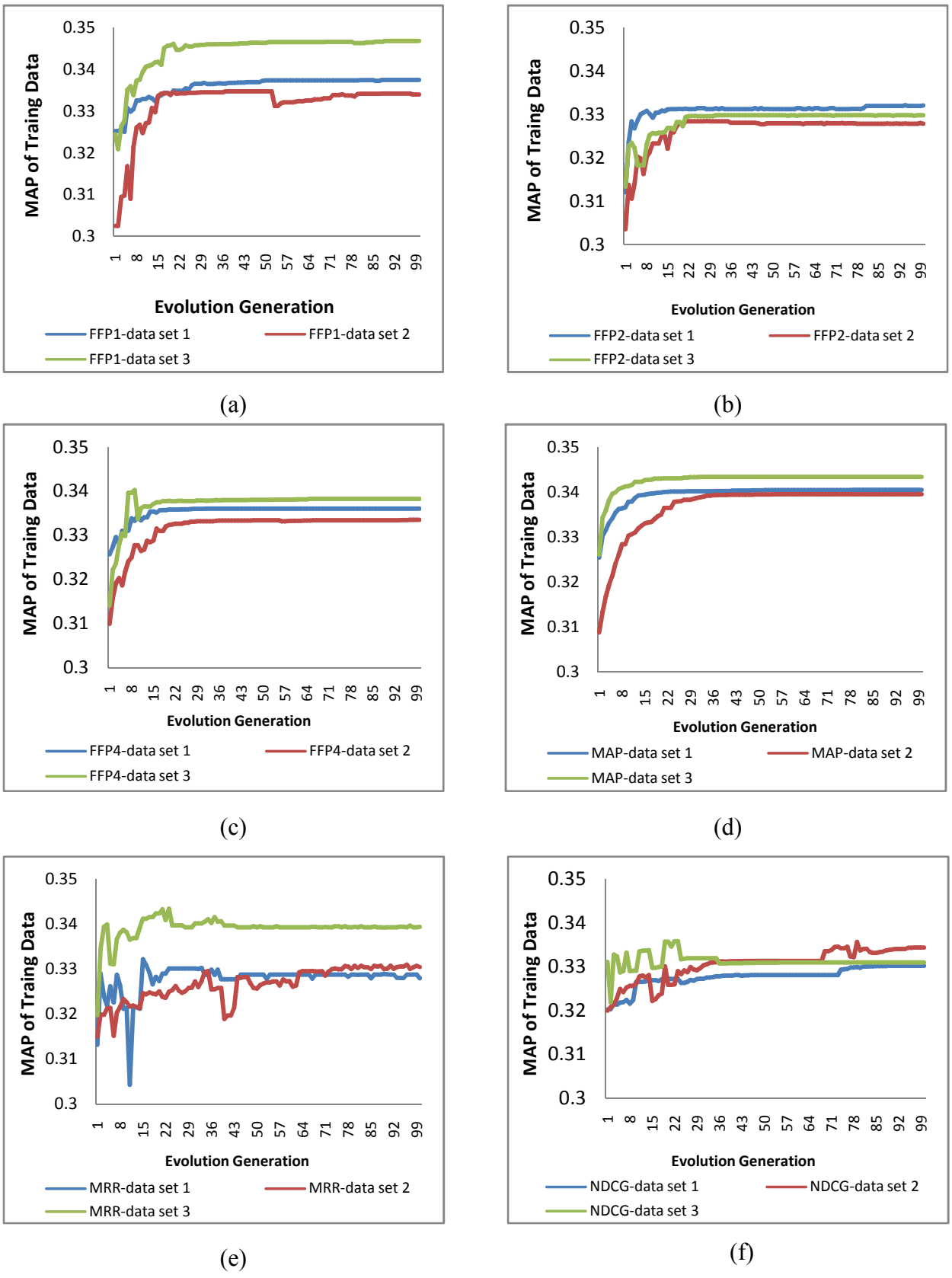
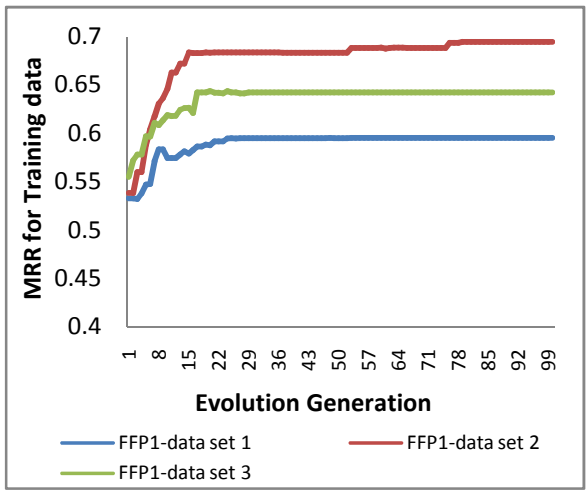
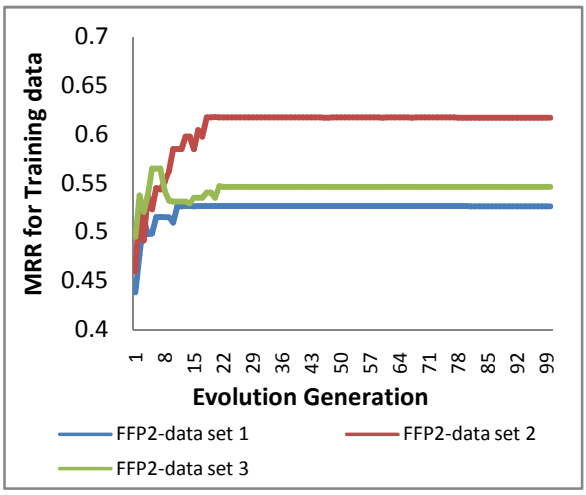


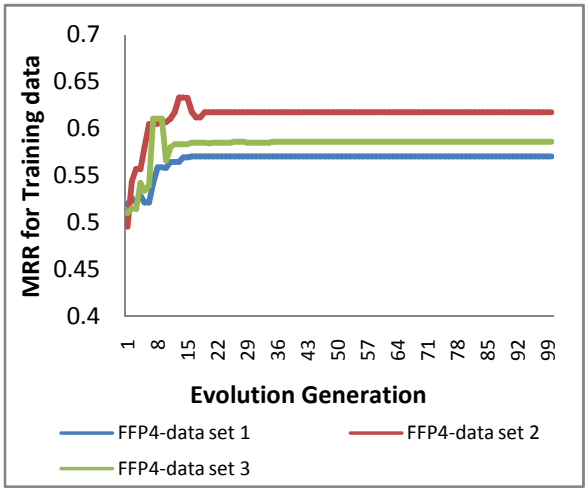
Figure 15 Training Improvement measured by MAP, by different fitness function, by different data set



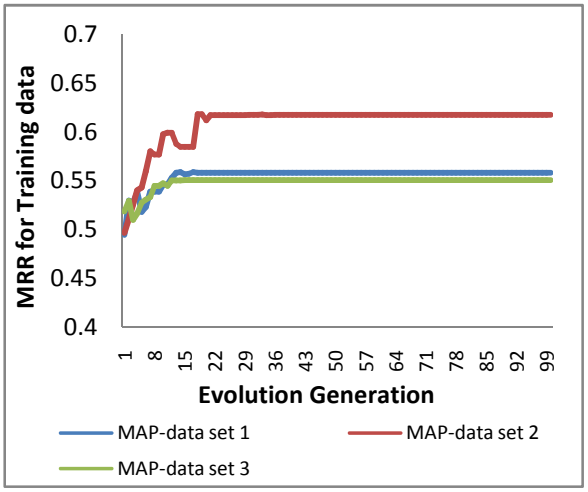
(a)



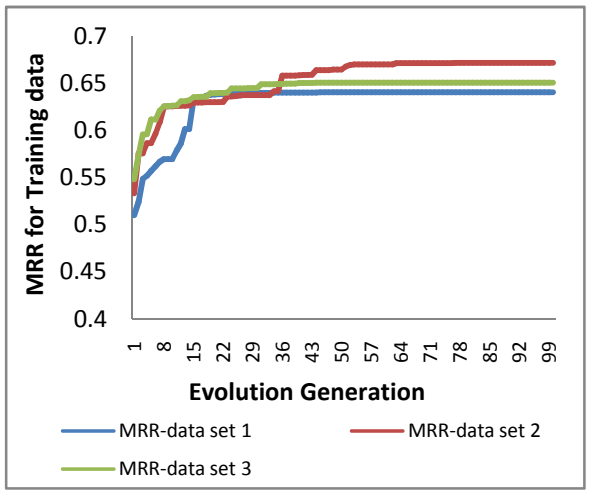
(b)



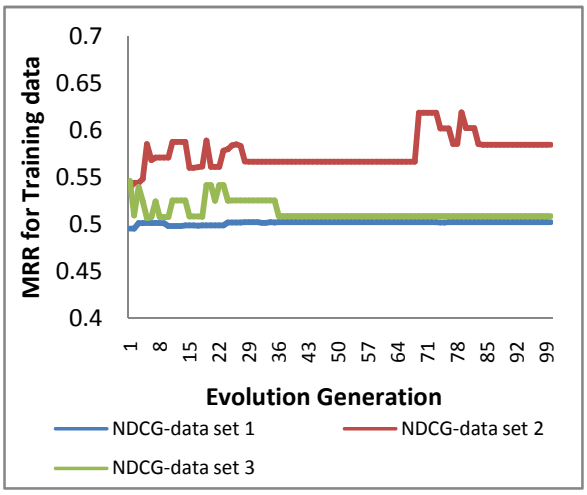
(c)



(d)



(e)



(f)

Figure 16 Training Improvement measured by MRR, by different fitness function, by different data set

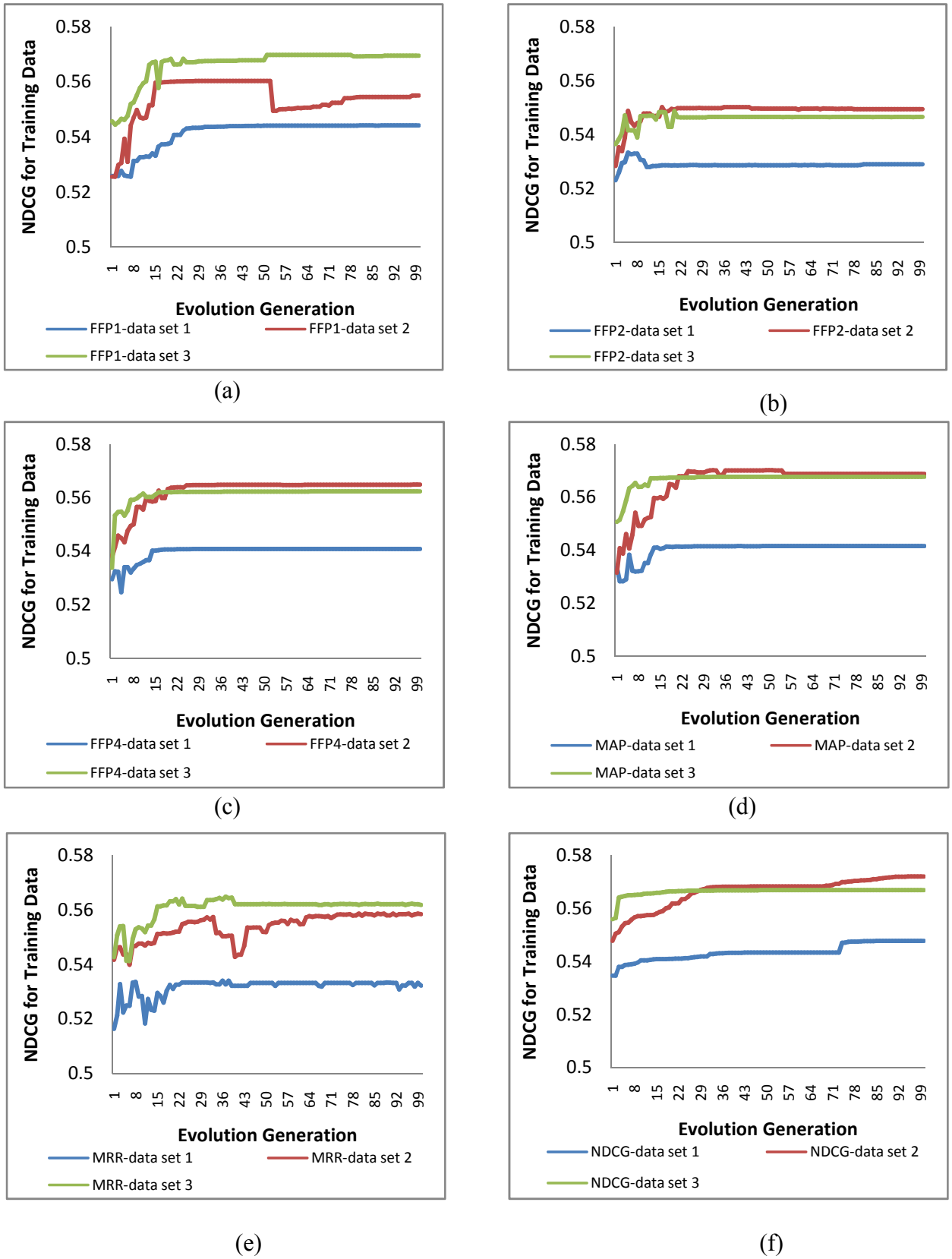
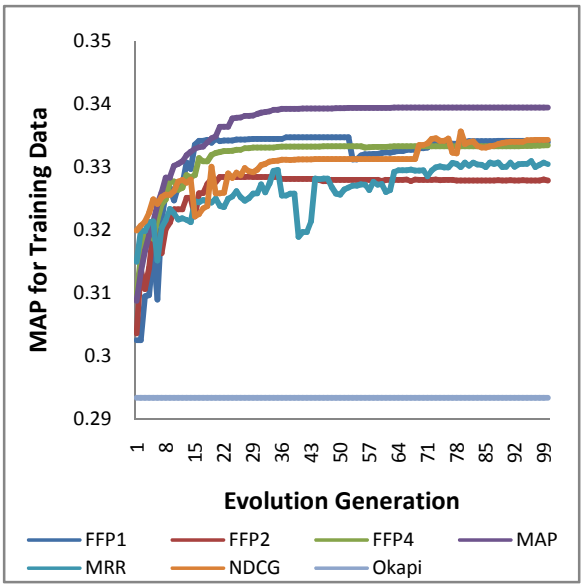
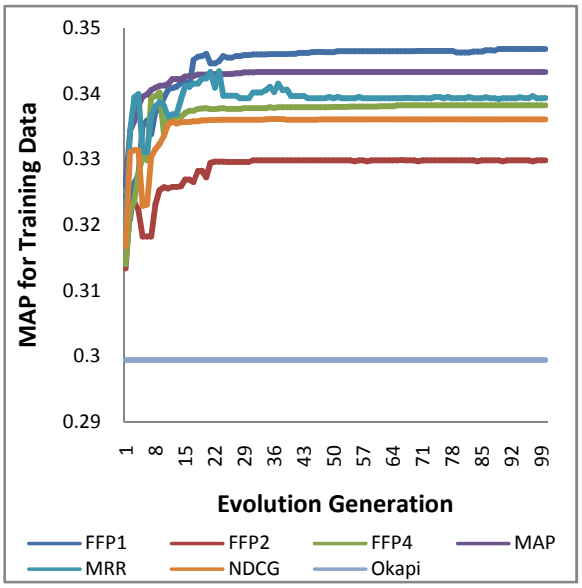


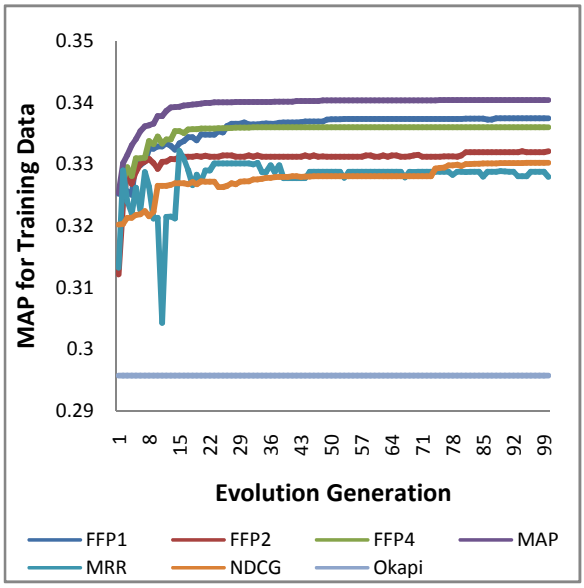
Figure 17 Training Improvement measured by NDCG, by different fitness function, by different data set



(a)

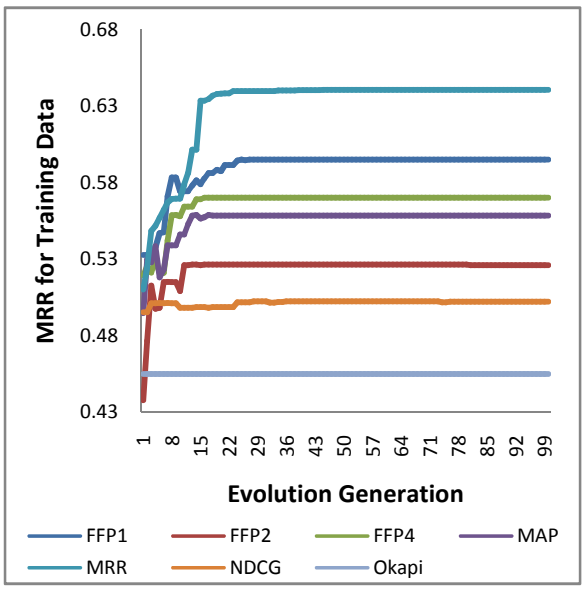


(b)

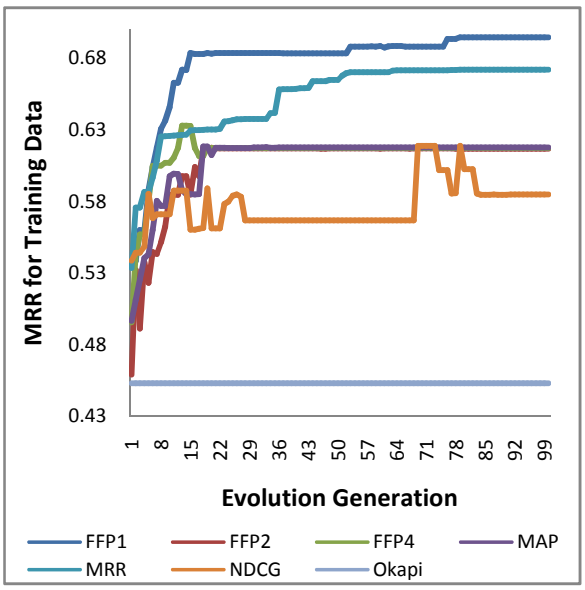


(c)

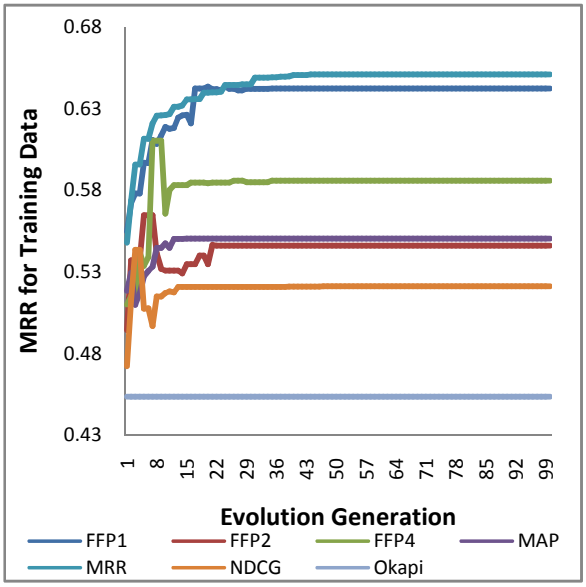
Figure 18 Training Improvement Measured by MAP– by different data set



(a)

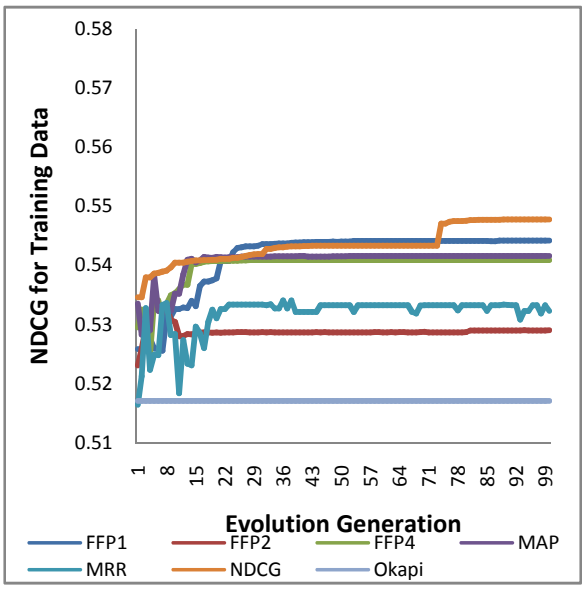


(b)

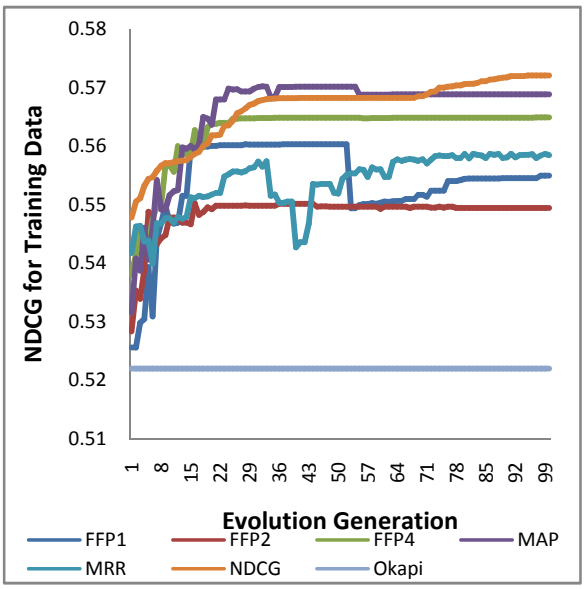


(c)

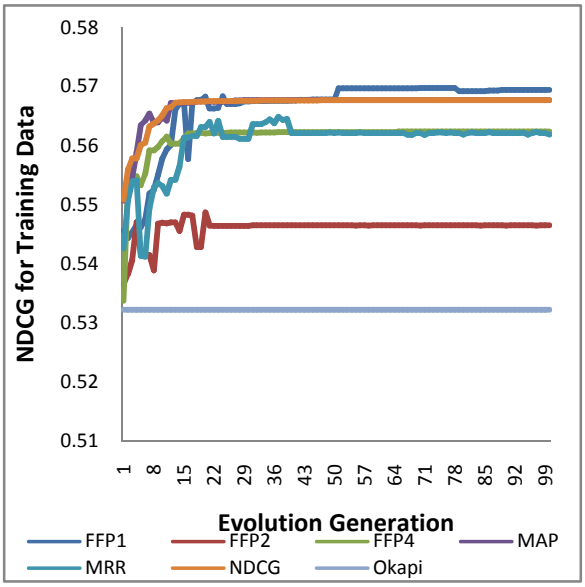
Figure 19 Training Improvement Measured by MRR – by different data set



(a)



(b)



(c)

Figure 20 Training Improvement Measured by MRR – by different data set

From the **Figures 14** we could learn that almost all the fitness function grow the fastest before the 15th generation, and then grow slower and slower till the value is convergent. We found two typical rules (except NDCG **Figure 14(f)**):

- In the early stage of the evolution, the fitness function value increases by every generation. However, in the later stage, the value stays the same for a number of generations, and then increase. Then the value will stay the same for larger number of generations before the next time of value increasing.
- The amount of the increased fitness value becomes smaller and smaller in the later stage of the evolution.

From the **Figure 15~17** and **Figure 18~20**, we could learn that training data performance measure value curve has very high correlation with the fitness function value curve. Value changes in the any fitness function always lead to the corresponding training data performance measure value changes, no matter MAP, MRR or NDCG. And it is obvious that training data performance measure value also increase fastest before the 15th generation.

5.2 Training Set Improvement Statistics Table

This section reports the statistics of the training data performance measures improvement.

	MAP -AVG	MAP -MAX	MAP -STV	RR -AVG	RR -MAX	RR -STV	NDCG -AVG	NDCG -MAX	NDCG -STV
FFP1 as fitness function									
data set 1	0.1454	0.1483	0.0030	0.3136	0.3692	0.0330	0.0464	0.0545	0.0079
data set 2	0.1497	0.1679	0.0137	0.5210	0.5418	0.0188	0.0702	0.0789	0.0071
data set 3	0.1560	0.1627	0.0063	0.3998	0.4160	0.0094	0.0630	0.0699	0.0040
AVG	0.1475	0.1581	0.0083	0.4173	0.4555	0.0259	0.0583	0.0667	0.0075
FFP2 as fitness function									
data set 1	0.1253	0.1344	0.0055	0.1711	0.1971	0.0218	0.0380	0.0435	0.0085
data set 2	0.1173	0.1256	0.0071	0.3086	0.3620	0.0344	0.0587	0.0672	0.0070
data set 3	0.1117	0.1258	0.0097	0.2497	0.3186	0.0427	0.0325	0.0452	0.0088
AVG	0.1181	0.1286	0.0074	0.2431	0.2926	0.0329	0.0431	0.0520	0.0081
FFP4 as fitness function									
data set 1	0.1353	0.1429	0.0079	0.2391	0.2535	0.0127	0.0459	0.0498	0.0026
data set 2	0.1347	0.1476	0.0088	0.3567	0.3629	0.0075	0.0788	0.0858	0.0058
data set 3	0.1331	0.1389	0.0079	0.2347	0.2918	0.0416	0.0597	0.0643	0.0036
AVG	0.1344	0.1431	0.0082	0.2768	0.3027	0.0206	0.0614	0.0666	0.0040
MAP as fitness function									
data set 1	0.1477	0.1511	0.0019	0.2459	0.2866	0.0357	0.0451	0.0475	0.0025
data set 2	0.1613	0.2013	0.0230	0.3642	0.4209	0.0333	0.0870	0.0930	0.0046
data set 3	0.1595	0.1780	0.0169	0.2995	0.3766	0.0719	0.0691	0.0789	0.0089
AVG	**0.1562**	**0.1768**	**0.0139**	0.3032	0.3614	**0.0470**	0.0670	0.0731	0.0054
MRR as fitness function									
data set 1	0.1204	0.1324	0.0151	0.3312	0.4082	0.0620	0.0352	0.0461	0.0084
data set 2	0.1233	0.1389	0.0115	0.5186	0.5416	0.0242	0.0634	0.0746	0.0084
data set 3	0.1336	0.1409	0.0094	0.4333	0.4458	0.0143	0.0559	0.0624	0.0107
AVG	0.1257	0.1374	0.0120	**0.4277**	**0.4652**	0.0335	0.0515	0.0610	**0.0092**
NDCG as fitness function									
data set 1	0.1228	0.1323	0.0100	0.1271	0.1435	0.0174	0.0577	0.0620	0.0041
data set 2	0.1397	0.1740	0.0222	0.2974	0.3770	0.0630	0.0911	0.1013	0.0093
data set 3	0.1202	0.1289	0.0088	0.1564	0.1778	0.0225	0.0671	0.0703	0.0021
AVG	0.1276	0.1450	0.0137	0.1936	0.2328	0.0343	**0.0720**	**0.0779**	0.0052

Table 3 Performance Statistics of three Training Data Sets

We could learn from the **Table 3** that when MAP is used to be the fitness function, the corresponding training data MAP value gets the biggest improvement. The similar consistent also happens when fitness function is MRR and NDCG.

5.3 Improvement by GA

In this section, we report the performance of our GA based ranking strategy. The data tables show the improvements percentages over the five types of baselines values.

"AQ only" is a two stage cascade ranking function. It selects top 50 results returned by Okapi, and re-ranks it by sum of all the AQ features value.

"SC only" is a two stage cascade ranking function. It selects top 50 results returned by Okapi, and re-ranks it by sum of all the SC features value.

"AQ SC equal" is also a two stage cascade ranking function. It selects top 50 results returned by Okapi, and re-ranks it by sum of all the AQ and SC features value. However, there are different numbers of AQ and SC features. When we calculate the sum of all the features, we re-assign AQ features and SC features equal weights by dividing sum of all AQ features and sum of all SC features by their corresponding features numbers.

"SC + AQ" is a three stage cascade ranking function. It selects top 50 results returned by Okapi, and re-rank it by sum of all the SC features value, then selects top N (N<=50) from the results, and re-rank them by sum of all the AQ features value. We fix N=10 in our experiment.

The **Table 8** shows the best trained weights associated with features when the GA fitness function is FFP2. It suggests the "importance" of features which are used to judge the document helpfulness.

		MAP_10		P_10		MRR_10		NDCG_10	
FFP1 as GA Fitness Function		0.1586		0.2846		0.5764		0.3179	
Baseline Functions	Okapi	0.1206	(31.55%)	0.2462	(15.63%)	0.4137	(39.33%)	0.2548	(24.75%)
	AQ only	0.1395	(13.73%)	0.2385	(19.35%)	0.4893	(17.79%)	0.2948	(7.83%)
	SC only	0.1016	(56.14%)	0.2231	(27.59%)	0.3782	(52.40%)	0.2316	(37.25%)
	AQ SC equal	0.1184	(34.01%)	0.2538	(12.12%)	0.4289	(34.37%)	0.2679	(18.67%)
	SC + AQ	0.1394	(13.77%)	0.2384	(19.38%)	0.4893	(17.80%)	0.2948	(7.84%)
FFP2 as GA Fitness Function		0.1293		0.2231		0.5374		0.2873	
Baseline Functions	Okapi	0.1206	(7.21%)	0.2462	(-9.38%)	0.4137	(29.91%)	0.2548	(12.73%)
	AQ only	0.1395	(-7.31%)	0.2385	(-6.45%)	0.4893	(9.83%)	0.2948	(-2.56%)
	SC only	0.1016	(27.25%)	0.2231	(-12.12%)	0.3782	(42.09%)	0.2316	(24.03%)
	AQ SC equal	0.1184	(9.22%)	0.2538	(-12.12%)	0.4289	(25.28%)	0.2679	(7.24%)
	SC + AQ	0.1394	(-7.24%)	0.2384	(-6.42%)	0.4893	(9.83%)	0.2948	(-2.54%)
FFP4 as GA Fitness Function		0.1377		0.2231		0.5521		0.2880	
Baseline Functions	Okapi	0.1206	(14.17%)	0.2462	(-9.38%)	0.4137	(33.47%)	0.2548	(13.01%)
	AQ only	0.1395	(-1.30%)	0.2385	(-6.45%)	0.4893	(12.84%)	0.2948	(-2.32%)
	SC only	0.1016	(35.51%)	0.2231	(0.00%)	0.3782	(45.99%)	0.2316	(24.34%)
	AQ SC equal	0.1184	(16.31%)	0.2538	(-12.12%)	0.4289	(28.72%)	0.2679	(7.50%)
	SC + AQ	0.1394	(-1.21%)	0.2384	(-6.42%)	0.4893	(12.83%)	0.2948	(-2.31%)
MAP as GA Fitness Function		0.1382		0.2538		0.5424		0.2858	
Baseline Functions	Okapi	0.1206		0.2462		0.4137		0.2548	
	AQ only	0.1395	(-0.89%)	0.2385	(6.45%)	0.4893	(10.85%)	0.2948	(-3.05%)
	SC only	0.1016	(36.07%)	0.2231	(13.79%)	0.3782	(43.41%)	0.2316	(23.41%)
	AQ SC equal	0.1184	(16.79%)	0.2538	(0.00%)	0.4289	(26.45%)	0.2679	(6.71%)
	SC + AQ	0.1394	(-0.86%)	0.2384	(6.46%)	0.4893	(10.85%)	0.2948	(-3.05%)
MRR as GA Fitness Function		0.160		0.2538		0.5923		0.3133	
Baseline Functions	Okapi	0.121	(32.55%)	0.2462	(3.13%)	0.4137	(43.18%)	0.2548	(22.94%)
	AQ only	0.139	(14.59%)	0.2385	(6.45%)	0.4893	(21.05%)	0.2948	(6.26%)
	SC only	0.102	(57.32%)	0.2231	(13.79%)	0.3782	(56.61%)	0.2316	(35.27%)
	AQ SC equal	0.118	(35.03%)	0.2538	(0.00%)	0.4289	(38.09%)	0.2679	(16.96%)
	SC + AQ	0.1394	(14.78%)	0.2384	(6.46%)	0.4893	(21.05%)	0.2948	(6.28%)
NDCG as GA Fitness Function		0.1394		0.2538		0.4538		0.2784	
Baseline Functions	Okapi	0.1206	(15.64%)	0.2462	(3.13%)	0.4137	(9.71%)	0.2548	(9.26%)
	AQ only	0.1395	(-0.03%)	0.2385	(6.45%)	0.4893	(-7.25%)	0.2948	(-5.57%)
	SC only	0.1016	(37.25%)	0.2231	(13.79%)	0.3782	(20.00%)	0.2316	(20.21%)
	AQ SC equal	0.1184	(17.80%)	0.2538	(0.00%)	0.4289	(5.81%)	0.2679	(3.93%)
	SC + AQ	0.1394	(0%)	0.2384	(6.46%)	0.4893	(-7.26%)	0.2948	(-5.56%)

Table 4 Improvement by GA Approach by data set 1

		MAP_10		P_10		MRR_10		NDCG_10	
FFP1 as GA Fitness Function		0.1584		0.2000		0.4282		0.2776	
Baseline Functions	Okapi	0.1547	(2.39%)	0.2308	(-13.33%)	0.4248	(0.80%)	0.2813	(-1.32%)
	AQ only	0.1041	(52.22%)	0.1692	(18.18%)	0.3314	(29.21%)	0.2024	(37.13%)
	SC only	0.1167	(35.71%)	0.1846	(8.33%)	0.4002	(7.00%)	0.2275	(22.01%)
	AQ SC equal	0.1222	(29.61%)	0.1846	(8.33%)	0.4112	(4.14%)	0.2420	(14.70%)
	SC + AQ	0.1041	(52.16%)	0.1692	(18.20%)	0.3314	(29.21%)	0.2024	(37.15%)
FFP12as GA Fitness Function		0.1833		0.2077		0.4981		0.2944	
Baseline Functions	Okapi	0.1547	(18.49%)	0.2308	(-10.00%)	0.4248	(17.25%)	0.2813	(4.67%)
	AQ only	0.1041	(76.16%)	0.1692	(22.73%)	0.3314	(50.29%)	0.2024	(45.45%)
	SC only	0.1167	(57.05%)	0.1846	(12.50%)	0.4002	(24.46%)	0.2275	(29.41%)
	AQ SC equal	0.1222	(49.99%)	0.1846	(12.50%)	0.4112	(21.14%)	0.2420	(21.65%)
	SC + AQ	0.1041	(76.08%)	0.1692	(22.75%)	0.3314	(50.30%)	0.2024	(45.45%)
FFP4 as GA Fitness Function		0.1626		0.2000		0.3892		0.2596	
Baseline Functions	Okapi	0.1547	(5.07%)	0.2308	(-13.33%)	0.4248	(-8.38%)	0.2813	(-7.73%)
	AQ only	0.1041	(56.21%)	0.1692	(18.18%)	0.3314	(17.44%)	0.2024	(28.23%)
	SC only	0.1167	(39.26%)	0.1846	(8.33%)	0.4002	(-2.75%)	0.2275	(14.09%)
	AQ SC equal	0.1222	(33.00%)	0.1846	(8.33%)	0.4112	(-5.35%)	0.2420	(7.25%)
	SC + AQ	0.1041	(56.19%)	0.1692	(18.20%)	0.3314	(17.44%)	0.2024	(28.26%)
MAR as GA Fitness Function		0.1530		0.2154		0.3859		0.2718	
Baseline Functions	Okapi	0.1547	(-1.12%)	0.2308	(-6.67%)	0.4248	(-9.15%)	0.2813	(-3.37%)
	AQ only	0.1041	(47.02%)	0.1692	(27.27%)	0.3314	(16.44%)	0.2024	(34.28%)
	SC only	0.1167	(31.06%)	0.1846	(16.67%)	0.4002	(-3.57%)	0.2275	(19.47%)
	AQ SC equal	0.1222	(25.17%)	0.1846	(16.67%)	0.4112	(-6.15%)	0.2420	(12.31%)
	SC + AQ	0.1041	(46.97%)	0.1692	(27.30%)	0.3314	(16.45%)	0.2024	(23.32%)
MRR as GA Fitness Function		0.1430		0.2000		0.3756		0.2577	
Baseline Functions	Okapi	0.1547	(-7.59%)	0.2308	(-13.33%)	0.4248	(-11.57%)	0.2813	(-8.41%)
	AQ only	0.1041	(37.40%)	0.1692	(18.18%)	0.3314	(13.35%)	0.2024	(27.28%)
	SC only	0.1167	(22.49%)	0.1846	(8.33%)	0.4002	(-6.13%)	0.2275	(13.24%)
	AQ SC equal	0.1222	(16.98%)	0.1846	(8.33%)	0.4112	(-8.64%)	0.2420	(6.46%)
	SC + AQ	0.1041	(37.37%)	0.1692	(18.20%)	0.3314	(13.34%)	0.2024	(27.32%)
NDCG as GA Fitness Function		0.1613		0.2154		0.4556		0.2695	
Baseline Functions	Okapi	0.1547	(4.23%)	0.2308	(-6.67%)	0.4248	(7.24%)	0.2813	(-4.20%)
	AQ only	0.1041	(54.97%)	0.1692	(27.27%)	0.3314	(37.46%)	0.2024	(33.12%)
	SC only	0.1167	(38.15%)	0.1846	(16.67%)	0.4002	(13.84%)	0.2275	(18.44%)
	AQ SC equal	0.1222	(31.94%)	0.1846	(16.67%)	0.4112	(10.79%)	0.2420	(11.34%)
	SC + AQ	0.1041	(54.95%)	0.1692	(27.30%)	0.3314	(37.48%)	0.2024	(38.98%)

Table 5 Improvement by GA Approach by data set 2

		MAP_10		P_10		MRR_10		NDCG_10	
FFP1 as GA Fitness Function		0.1904		0.2538		0.5161		0.3062	
Baseline Functions	Okapi	0.1624	(17.21%)	0.2154	(17.86%)	0.4278	(20.65%)	0.2813	(8.83%)
	AQ only	0.1308	(45.60%)	0.2077	(22.22%)	0.3234	(59.60%)	0.2492	(22.87%)
	SC only	0.1041	(82.91%)	0.1769	(43.48%)	0.3397	(51.91%)	0.1961	(56.14%)
	AQ SC equal	0.0963	(97.67%)	0.1846	(37.50%)	0.3071	(68.04%)	0.2043	(49.89%)
	SC + AQ	0.1307	(45.48%)	0.2076	(22.25%)	0.3233	(59.64%)	0.2491	(22.92%)
FFP2 as GA Fitness Function		0.2192		0.2538		0.6079		0.3514	
Baseline Functions	Okapi	0.1624	(34.93%)	0.2154	(17.86%)	0.4278	(42.11%)	0.2813	(24.89%)
	AQ only	0.1308	(67.62%)	0.2077	(22.22%)	0.3234	(87.98%)	0.2492	(41.01%)
	SC only	0.1041	(110.57%)	0.1769	(43.48%)	0.3397	(78.93%)	0.1961	(79.18%)
	AQ SC equal	0.0963	(127.56%)	0.1846	(37.50%)	0.3071	(97.92%)	0.2043	(72.01%)
	SC + AQ	0.1307	(67.71%)	0.2076	(22.25%)	0.3233	(88.03%)	0.2491	(41.68%)
FFP4 as GA Fitness Function		0.1987		0.2462		0.5418		0.3199	
Baseline Functions	Okapi	0.1624	(22.33%)	0.2154	(14.29%)	0.4278	(26.64%)	0.2813	(13.71%)
	AQ only	0.1308	(51.96%)	0.2077	(18.52%)	0.3234	(67.53%)	0.2492	(28.38%)
	SC only	0.1041	(90.90%)	0.1769	(39.13%)	0.3397	(59.46%)	0.1961	(63.14%)
	AQ SC equal	0.0963	(106.30%)	0.1846	(33.33%)	0.3071	(76.39%)	0.2043	(56.61%)
	SC + AQ	0.1307	(54.03%)	0.2076	(18.59%)	0.3233	(67.58%)	0.2491	(28.42%)
MAP as GA Fitness Function		0.1931		0.2538		0.5154		0.3130	
Baseline Functions	Okapi	0.1624	(18.85%)	0.2154	(17.86%)	0.4278	(20.48%)	0.2813	(11.25%)
	AQ only	0.1308	(47.63%)	0.2077	(22.22%)	0.3234	(59.37%)	0.2492	(25.61%)
	SC only	0.1041	(85.47%)	0.1769	(43.48%)	0.3397	(51.70%)	0.1961	(59.62%)
	AQ SC equal	0.0963	(100.43%)	0.1846	(37.50%)	0.3071	(67.80%)	0.2043	(53.23%)
	SC + AQ	0.1307	(47.96%)	0.2076	(22.25%)	0.3233	(59.42%)	0.2491	(25.65%)
MRR as GA Fitness Function		0.2039		0.2538		0.5656		0.3409	
Baseline Functions	Okapi	0.1624	(25.49%)	0.2154	(17.86%)	0.4278	(32.22%)	0.2813	(21.18%)
	AQ only	0.1308	(55.88%)	0.2077	(22.22%)	0.3234	(74.90%)	0.2492	(36.82%)
	SC only	0.1041	(95.83%)	0.1769	(43.48%)	0.3397	(66.48%)	0.1961	(73.86%)
	AQ SC equal	0.0963	(111.63%)	0.1846	(37.50%)	0.3071	(84.15%)	0.2043	(66.90%)
	SC + AQ	0.1307	(56.01%)	0.2076	(22.25%)	0.3233	(74.95%)	0.2491	(36.85%)
NDCG as GA Fitness Function		0.1977		0.2538		0.5308		0.3192	
Baseline Functions	Okapi	0.1624	(21.73%)	0.2154	(17.86%)	0.4278	(24.08%)	0.2813	(13.46%)
	AQ only	0.1308	(51.21%)	0.2077	(22.22%)	0.3234	(64.13%)	0.2492	(28.11%)
	SC only	0.1041	(89.97%)	0.1769	(43.48%)	0.3397	(56.23%)	0.1961	(62.79%)
	AQ SC equal	0.0963	(105.29%)	0.1846	(37.50%)	0.3071	(72.81%)	0.2043	(56.28%)
	SC + AQ	0.1307	(51.26%)	0.2076	(22.25%)	0.3233	(64.18%)	0.2491	(28.14%)

Table 6 Improvement by GA approach by data set 3

		MAP_10		P_10		MRR_10		NDCG_10	
FFP1 as GA Fitness Function		0.1691		0.2462		0.5069		0.3006	
Baseline Functions	Okapi	0.1459	(15.90%)	0.2308	(6.67%)	0.4221	(20.09%)	0.2725	(10.31%)
	AQ only	0.1248	(35.50%)	0.2051	(20.04%)	0.3814	(32.91%)	0.2488	(20.82%)
	SC only	0.1075	(57.30%)	0.1949	(26.32%)	0.3727	(36.01%)	0.2184	(37.64%)
	AQ SC equal	0.1123	(50.58%)	0.2077	(18.54%)	0.3824	(32.56%)	0.2381	(26.25%)
	SC + AQ	0.1247	(35.61%)	0.2051	(20.04%)	0.3813	(32.94%)	0.2488	(20.82%)
FFP2 as GA Fitness Function		0.1773		0.2282		0.5478		0.3110	
Baseline Functions	Okapi	0.1459	(21.52%)	0.2308	(-1.13%)	0.4221	(29.78%)	0.2725	(14.13%)
	AQ only	0.1248	(42.07%)	0.2051	(11.26%)	0.3814	(43.63%)	0.2488	(25.00%)
	SC only	0.1075	(64.93%)	0.1949	(17.09%)	0.3727	(46.98%)	0.2184	(42.40%)
	AQ SC equal	0.1123	(57.88%)	0.2077	(9.87%)	0.3824	(43.25%)	0.2381	(30.62%)
	SC + AQ	0.1247	(42.18%)	0.2051	(11.26%)	0.3813	(43.67%)	0.2488	(25.00%)
FFP4 as GA Fitness Function		0.1663		0.2231		0.4944		0.2891	
Baseline Functions	Okapi	0.1459	(13.98%)	0.2308	(-3.34%)	0.4221	(17.13%)	0.2725	(6.09%)
	AQ only	0.1248	(33.25%)	0.2051	(8.78%)	0.3814	(29.63%)	0.2488	(16.20%)
	SC only	0.1075	(54.70%)	0.1949	(14.47%)	0.3727	(32.65%)	0.2184	(32.37%)
	AQ SC equal	0.1123	(48.09%)	0.2077	(7.41%)	0.3824	(29.29%)	0.2381	(21.42%)
	SC + AQ	0.1247	(33.36%)	0.2051	(8.78%)	0.3813	(29.66%)	0.2488	(16.20%)
MAP as GA Fitness Function		0.1614		0.2410		0.4812		0.2902	
Baseline Functions	Okapi	0.1459	(10.62%)	0.2308	(4.42%)	0.4221	(14.00%)	0.2725	(6.50%)
	AQ only	0.1248	(29.33%)	0.2051	(17.50%)	0.3814	(26.17%)	0.2488	(16.64%)
	SC only	0.1075	(50.14%)	0.1949	(23.65%)	0.3727	(29.11%)	0.2184	(32.88%)
	AQ SC equal	0.1123	(43.72%)	0.2077	(16.03%)	0.3824	(25.84%)	0.2381	(21.88%)
	SC + AQ	0.1247	(29.43%)	0.2051	(17.50%)	0.3813	(26.20%)	0.2488	(16.64%)
MRR as GA Fitness Function		0.1689		0.2359		0.5112		0.3040	
Baseline Functions	Okapi	0.1459	(15.76%)	0.2308	(2.21%)	0.4221	(21.11%)	0.2725	(11.56%)
	AQ only	0.1248	(35.34%)	0.2051	(15.02%)	0.3814	(34.03%)	0.2488	(22.19%)
	SC only	0.1075	(57.12%)	0.1949	(21.04%)	0.3727	(37.16%)	0.2184	(39.19%)
	AQ SC equal	0.1123	(50.40%)	0.2077	(13.58%)	0.3824	(33.68%)	0.2381	(27.68%)
	SC + AQ	0.1247	(35.45%)	0.2051	(15.02%)	0.3813	(34.07%)	0.2488	(22.19%)
NDCG as GA Fitness Function		0.1662		0.2410		0.4801		0.2890	
Baseline Functions	Okapi	0.1459	(13.91%)	0.2308	(4.42%)	0.4221	(13.74%)	0.2725	(6.06%)
	AQ only	0.1248	(33.17%)	0.2051	(17.50%)	0.3814	(25.88%)	0.2488	(16.16%)
	SC only	0.1075	(54.60%)	0.1949	(23.65%)	0.3727	(28.82%)	0.2184	(32.33%)
	AQ SC equal	0.1123	(48.00%)	0.2077	(16.03%)	0.3824	(25.55%)	0.2381	(21.38%)
	SC + AQ	0.1247	(33.28%)	0.2051	(17.50%)	0.3813	(25.91%)	0.2488	(16.16%)

Table 7 Overall improvement by GA Approach

Category	Features	Weight
AQ: Relevance	Okapi	0.1143
AQ: Information Richness	Length	0.7923
	Number of articles	0.7995
	Number of replies	0.5055
	Number of initial user replies	0.9079
AQ: Completeness	Number of users	0.2442
	Post per user	0.9798
AQ: Timeliness	Thread duration	0.6859
	Posting frequency	0.0639
	First response time	0.2722
	Last response time	0.3966
SC: Perceived expertise	Activity Length	0.0056
	Knowledge sharing activity level	0.4529
	Involvement	0.0256
	Knowledge scope/ broadness	0.0228
	Number of posts	0.7655
	Posting Frequency	0.7213
	Number of questions asked	0.0015
	Number of answer posts replies	0.3814
	Number of questions answered	0.0358
SC: Perceived Trustworthiness	Betweenness	0.0876
	Closeness	0.9028
	Clustering coefficient	0.0928
	In-degree	0.1389
	Out-degree	0.1351
	User rating	0.4385

Table 8 A features weights example -- the best trained chromosome by FFP2

5.4 Analysis and Discussion

There are several observations can be drawn after examining the results in above tables:

(1) The section 5.3 indicates that the data set 3 has the best training improvement. It is consistent with the fact that data set 3 gets the best performance measure improvement.

(2) From the **Table 7**, we can conclude the order of the effect of different fitness function by different performance measures:

MAP as performance measure function:

FFP2 > **FFP1** > **MRR** > NDCG > FFP4 > MAP

P as performance measure function:

FFP1 > NDCG = MAP > **MRR** > **FFP2** > FFP4

MRR as performance measure function:

FFP2 > **MRR** > **FFP1** > FFP4 > MAP > NDCG

NDCG as performance measure function:

FFP2 > **MRR** > **FFP1** > MAP > FFP4 > NDCG

We can say that FFP2, MRR, and FFP1 are the better reliable GA fitness functions than others, in the context of the OKC.

(3) From the **Table 8**, we can conclude that there are more "important" features in AQ category than in SC category. It also matched the results in the **Table 4~7** that normally ranking function "AQ only" gets more favorable results than ranking function "SC only".

5.6 Conclusion

In our study, we proposed a Genetics Algorithm based ranking optimization strategy to search authority documents in the Online Knowledge Community. Our model combines two sets of search criteria we discovered based on Knowledge Adoption Model: Argument Quality and Source Credibility. Prior to this work, content based or link based search functions are used by the Online Knowledge Community, which cannot bring good user search satisfaction. The major contribution of this work in the area of information retrieval is combination of Genetics Algorithm and Knowledge Adoption Model to build an optimized search strategy specifically for the Online Knowledge Community. The results solid proved that the retrieval performance of our system could beat all the baselines ranking functions. Here we also answered our research questions belonging to the early section of the paper.

- Our system works well for improving both Top N based retrieval measure function like P, and order-based retrieval measure functions like MAP, MRR, and NDCG.
- The performance of our system is stable. It consistently improves the baseline in all the three data sets. And it obviously beats the Okapi BM 25, which is a well known content based ranking function.
- Our system is independent from any specific argument quality feature or source credibility feature, which also makes our system flexible to extend if we discover more features by other methods.

- The experiment results suggest the different effect of different fitness functions. It proved that using utility theory is a good way to discover new fitness functions when Genetics Algorithm applies to the area of information retrieval.

Chapter 6

Future work

There are several interesting places for our further research.

- The current source credibility features assign user a set of static scores. However, it is a common sense that innovation of techniques requires people to continually learn the new knowledge. So, it is possible that a domain expert will become a newbie if he/she doesn't update the old knowledge frequently. Hence, we could identify the user expertise (source credibility) more precisely if we consider it in a time sensitive manner.
- In the chapter 1, we discussed that because of the absence of hyperlinks among threads in the online forums, link based ranking algorithm cannot be directly used in the OKC. However, it will be interesting if we could apply link based algorithm, like PageRank, to the user social network. It is unclear whether there exist similar PageRank concepts in social network as the ones in the web pages network.
- We currently only use linear combination of features in the Genetics Algorithm. It will be interesting if we could use the polynomial functions formats to combine those features.
- Although the results confirm that our utility functions have quite promising performance, it is entirely possible that there exist better utility functions, whose curve shape is closer to the shape of the ideal utility function.

- Currently we have 43 queries in total. Re-exam the theory we got with larger number of queries will be helpful to prove our model to be a general working ranking optimization strategy in the Online Knowledge Community.
- We currently treat the social network as an un-weighted network. It will be interesting if we re-calculate the Clustering Coefficient value and re-run the experiment, by treating the social network as a weighted network.

References

1. V. Batagelj, "Social network analysis: Methods and applications," *Psychometrika*, vol. 63, no. 1, 1998, pp. 103-104.

2. A. Singhal, G. Salton, M. Mitra, and C. Buckley, "Document length normalization," *Information Processing & Management*, vol. 32, no. 5, 1996, pp. 619-633.

3. S.W. S. Robertson, and M. Hancock-Beaulieu, "Experimentation as a way of life: Okapi at TREC," *Information Processing and Management*, vol. 36(1):95–108, 2000.

4. S. Brin, and L. Page, "The anatomy of a large-scale hypertextual Web search engine," *7th International World Wide Web Conference*, pp. 107-117.

5. J. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the Acm*, vol. 46:604–632, 1999.

6. J. Nahapiet, and S. Ghoshal, "Social capital, intellectual capital, and the organizational advantage," *Academy of Management Review*, vol. 23, no. 2, 1998, pp. 242-266.

7. M.S. Ackerman, "Answer Garden - a Tool for Growing Organizational Memory," *Wirtschaftsinformatik*, vol. 37, no. 3, 1995, pp. 320-321.

8. M.S. Ackerman, and D.W. McDonald, "Answer Garden 2: merging organizational memory with collaborative help," *Book Answer Garden 2: merging organizational memory with collaborative help*, Series Answer Garden 2: merging organizational memory with collaborative help, ed., Editor ed.^eds., ACM, 1996, pp.

9. B. Krulwich, C. Burkey, and A.A.A.I. Amer Assoc Artificial Intelligence, "The ContactFinder agent: Answering bulletin board questions with referrals," *13th National Conference on Artificial Intelligence (AAAI 96) / 8th Conference on Innovative Applications of Artificial Intelligence (IAAI 96)*, Amer Assoc Artificial Intelligence, pp. 10-15.

10. P. Gray, "Sharing expertise: Beyond knowledge management," *Information Systems Management*, vol. 22, no. 1, 2005, pp. 91-94.

11. H. Kautz, B. Selman, and M. Shah, "Referral web: Combining social networks and collaborative filtering," *Communications of the Acm*, vol. 40, no. 3, 1997, pp. 63-65.

12. F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 9, 2004, pp. 2658-2663.

13. G. Abramson, and M. Kuperman, "Social games in a social network," *Physical Review E*, vol. 63, no. 3, 2001.

14. M. Girvan, and M.E.J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, 2002, pp. 7821-7826.

15. R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt, and A. Arenas, "Self-similar community structure in a network of human interactions," *Physical Review E*, vol. 68, no. 6, 2003.

16. M.E.J. Newman, "Detecting community structure in networks," *European Physical Journal B*, vol. 38, no. 2, 2004, pp. 321-330.

17. M.E.J. Newman, and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, 2004.

18. Z.B. Kou, and C.S. Zhang, "Reply networks on a bulletin board system," *Physical Review E*, vol. 67, no. 3, 2003.

19. C.M. Chiu, M.H. Hsu, and E.T.G. Wang, "Understanding knowledge sharing in virtual communities: An integration of social capital and social cognitive theories," *Decision Support Systems*, vol. 42, no. 3, 2006, pp. 1872-1888.

20. W.V. Y. Li, W. Schoenmakers, D. Agoralaan-building, and D. Gebouw, "Exploration and Exploitation in Innovation: Reframing the Interpretation," *Creativity and Innovation Management*, vol. vol. 17, pp. 107-126, 2008.

21. M. Smith, "Tools for navigating large social cyberspaces," *Commun. ACM*, vol. 45(4):51–55, 2002.

22. W. Sack, "Discourse diagrams: Interface design for very large-scale conversations," *HICSS'00*, vol. Volume 3, page 3034, no. IEEE CS, 2000.

23. S. Whittaker, L. Terveen, W. Hill, and L. Cherny, "The dynamics of mass interaction," *Book The dynamics of mass interaction*, Series The dynamics of mass interaction, ed., Editor ed.^eds., ACM, 1998, pp.

24. M.A. Smith, "Netscan: Measuring and Mapping the Social Structure of Usenet," *International Sunbelt Social Network*, 1997.

25. M. Smith, "*Netscan: A Tool for Measuring and Mapping Social Cyberspaces*," 2001.

26. M.A. Smith, "Measures and Maps of Usenet, in From Usenet to Cowebs: Interacting with Social Information," *Springer Verlag:*, vol. p. 47-78., no. Amsterdam, Holland, 2003.

27. M.A. Smith, "Netscan: A Social Accounting Search Engine," *Community Technologies Group, Microsoft Corporation*, 2004.

28. D.G. N. Matsumura, and X. Llora, "Mining directed social network from message board," *14th WWW '05*, vol. pages 1092–1093, no. ACM Press, 2005.

29. K.I. Goh, Y.H. Eom, H. Jeong, B. Kahng, and D. Kim, "Structure and evolution of online social relationships: Heterogeneity in unrestricted discussions," *Physical Review E*, vol. 73, no. 6, 2006.

30. D. Maloney-Krichmar, and J. Preece, "A multilevel analysis of sociability, usability, and community dynamics in an online health community," *ACM Trans. Comput.-Hum. Interact.*, vol. 12, no. 2, 2005, pp. 201-232.

31. Vicen\, \#231, G\, \#243, mez, A. Kaltenbrunner, V. L\, and pez, "Statistical analysis of the social network and discussion threads in slashdot," *Book Statistical analysis of the social network and discussion threads in slashdot*, Series Statistical analysis of the social network and discussion threads in slashdot, ed., Editor ed.^eds., ACM, 2008, pp.

32. S.W. A. Vilpponen, and S. Sundqvist, "Electronic Word-of-Mouth in Online Environments: Exploring Referral Network Structure and Adoption Behavior," *Journal of Interactive Advertising*, vol. vol. 6, pp. 71-86, 2006.

33. C.W.a.P. Morrison, "Network Analysis in Marketing," *Australasian Marketing Journal*, vol. vol. 12, pp. 8-18, 2004.

34. M.K. A. Mehra, and D. Brass, "The Social Networks of High and Low Self-Monitors: Implications for Workplace Performance," *Administrative Science Quarterly*, vol. pp. 121-146, 2001.

35. M.E.J. Newman, "Scientific Collaboration Networks. Ii. Shortest Paths, Weighted Networks, and Centrality," *Physical Review E*, vol. vol. 64, p. 016132, 2001.

36. R.E. Petty, and Cacioppo, J.T, "*Communication and Persuasion: Central and Peripheral Routes to Attitude Change*," *New York: Springer-Verlag*, 1986.

37. S.W. Sussman, and Siegal, W.S., "Informational Influence in Organizations: An Integrated Approach to Knowledge Adoption," *Information Systems Research*, vol. pp 47-65, 2003.

38. D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Co. , 1989.

39. H. Xie, M. Zhang, and P. Andreae, "An analysis of constructive crossover and selection pressure in genetic programming," *Book An analysis of constructive crossover and selection pressure in genetic programming*, Series An analysis of constructive crossover and selection pressure in genetic programming, ed., Editor ed.^eds., ACM, 2007, pp.

40. H.C. Chen, G. Shankaranarayanan, L.L. She, and A. Iyer, "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, 1998, pp. 693-705.

41. M.P. Smith, and M. Smith, "The use of genetic programming to build Boolean queries for text retrieval through relevance feedback," *Journal of Information Science*, vol. 23, no. 6, 1997, pp. 423-431.

42. J.T. Horng, and C.C. Yeh, "Applying genetic algorithms to query optimization in document retrieval," *Information Processing & Management*, vol. 36, no. 5, 2000, pp. 737-759.

43. J.J. Yang, and R.R. Korfhage, "Query Optimization in Information-Retrieval Using Genetic Algorithms," *Proceedings of the Fifth International Conference on Genetic Algorithms*, 1993, pp. 603-611

665.

44. W. Fan, M.D. Gordon, and P. Pathak, "Personalization of search engine services for effective retrieval and knowledge management," *Book Personalization of search engine services for effective retrieval and knowledge management*, Series Personalization of search engine services for effective retrieval and knowledge management, ed., Editor ed.^eds., Association for Information Systems, 2000, pp.

45. W.G. Fan, M.D. Gordon, and P. Pathak, "Discovery of context-specific ranking functions for effective information retrieval using genetic programming," *Ieee Transactions on Knowledge and Data Engineering*, vol. 16, no. 4, 2004, pp. 523-527.

46. W.G. Fan, M.D. Gordon, and P. Pathak, "A generic ranking function discovery framework by genetic programming for information retrieval," *Information Processing & Management*, vol. 40, no. 4, 2004, pp. 587-602.

47. C. Lopez-Pujalte, and V.P. Guerrero-Bote, "Order-based fitness functions for genetic algorithms applied to relevance feedback," *Journal of the American Society for Information Science and Technology*, vol. 54, no. 2, 2003, pp. 152-160.

48. W.G. Fan, E.A. Fox, P. Pathak, and H. Wu, "The effects of fitness functions on genetic programming-based ranking discovery for web search," *Journal of the American Society for Information Science and Technology*, vol. 55, no. 7, 2004, pp. 628-636.

49. O. Kurland, and Lee, L, "Pagerank without Hyperlinks: Structural Re-Ranking Using Links Induced by Language Models," *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,*, vol. pp. 306-313, 2005.

50. M. Heesacker, Petty, R.E., and Cacioppo, J.T., "Field Dependence and Attitude Change: Source Credibility Can Alter Persuasion by Affecting Message Relevant Thinking," *Journal of Personality and Social Psychology*, vol. pp 653-666, 1983.