SEMANTIC SWARM INTELLIGENCE FOR CANDIDATE LINKS

Ms.Susan Geethu.D.K[1], Ms. R.Subha[2], Dr.S.Palaniswami[3]

[1]PG Scholar, [2]Assistant Professor
[1,2]Department of Computer Science and Engineering, Sri Krishna College of Technology, Coimbatore, India
[1]geethudaniel@gmail.com, [2]kris.subha@gmail.com
[3]Principal, [3]Government College of Engineering,
Bargur, Krishnagiri, India
[3]joegct81@yahoo.com

ABSTRACT

Requirements traceability is an important activity undertaken as part of ensuring the quality of software in the early stages of the Software Development Life Cycle (SDLC). Requirements tracing of natural Language artifacts consists of document parsing, Candidate Link Generation, evaluation and analysis. Candidate Link Generation deals with checking if the high-level artifact has been fulfilled by the low-level artifact. The Candidate Link can be established using Swarm Techniques which generates Requirements Traceability Matrices (RTMs) between textual requirements artifacts (high level requirements traced to low level requirements, for example) with better accuracy than traditional information retrieval techniques. The Semantic Relatedness between the terms is not considered in the existing system; hence the Candidate Link Generation is not effective. In the proposed system, a hybrid technique combining both the Semantic Ranking and Pheromone Swarm is implemented. Simple swarm agents are given freedom to operate on their own, determining the search path randomly based on the environment. Pheromone swarm agent decides on what term to select or what path to take is influenced by presence of pheromone markings on the inspected object. Semantic Graph is constructed using semantic relatedness between two terms, computed based on highest value path connecting any pair of the terms. The performance is evaluated with Simple, Pheromone and Semantic Pheromone Swarm techniques. The Semantic Pheromone Swarm provides better results when compared to Simple and Pheromone Swarm Techniques.

KEYWORDS

Information Retrieval, Requirements Traceability, Semantic Rank, Software Engineering & Swarms

1. INTRODUCTION

Requirement collection plays the major role, in developing a project. If requirements are captured, they may not be formally documented, analyzed, kept up to date or traced as a software development life cycle progresses. The lack of formal requirements or lack of quality requirements leads to poor software quality. Activities are undertaken to improve the quality of requirements including requirements consistency checking, requirements tracing etc.These techniques may be more computationally complex than other techniques that have been widely applied in requirements engineering (such as information retrieval techniques for requirements tracing). Some of these activities have been supported by automated techniques. These techniques though are not fully automatic, are not general-purpose, have not been validated on large, real-world systems in numerous domains, and still require much effort on the part of human analyst.

DOI : 10.5121/ijci.2012.1301

As the result, researchers continue to search for new and better techniques to improve the quality of requirement. Candidate link generation is concerned with retrieving relevant information [4]. Swarm algorithm is used to rank retrieved low-level requirement elements that may be relevant to high-level requirements which is adapted for candidate link generation. Researchers have successfully applied Swarm Techniques to a number of problems in software maintenance.

1.1 Swarm Intelligence

Swarm Intelligence (SI) is the collective behavior of decentralized, self-organized systems, natural or artificial. SI systems typically employ a relatively large population of agents, which interact both between themselves and the environment [4]. As a result, all these agents, being simple, unable to reason a decision just by themselves, share knowledge and optimize their behavior on the basis of the knowledge shared.

1.2 Ant Colony Optimization

The most important fact about ant colonies is that they base their choice of path to a food source completely on the pheromone level. At first, when they start searching for a food source, they simply wander around. After an ant finds a food source, on its way back to the nest, it lays down a pheromone trail, which increases the possibility of the fellow ants locating the food source. The fellow ants then follow the pheromone trace [4] and if they locate the food source, on their way back to the nest they lay down pheromone trails as well. If they did not find the food source, or the pheromone trail is far from optimal and it takes the ants too much time to get back to the nest, the pheromone trails evaporate, giving an opportunity to the other ants to locate the same food source via shorter path. Fig 1 represents the behavior of ants when they are looking for food. In the first case, an ant has just located a food source and left a pheromone trail to this food source.

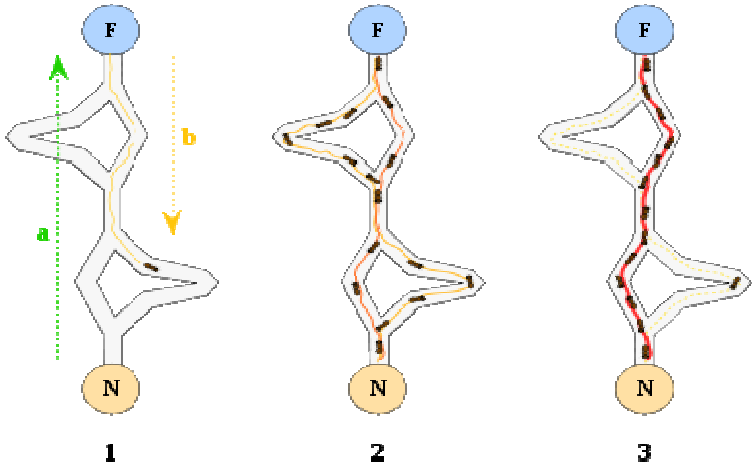

Figure 1. Ant colony behavior

The second case shows the moment when lots of ants have already discovered the food source, but using different paths. Finally, the third case shows that at some point most ants are using the "optimal" path. Actually, the path might still not be optimal, but it should be rather close to

optimal if enough agents are present. That is why this algorithm is perfectly suitable for NP-complete problems such as the travelling salesman one.

1.3 Semantic Ranking

Semantic relatedness between two terms can be computed based on highest value path connecting any pair of the terms [3]. In finding highest value, the different meanings (senses) that appear between each word are determined. High Level document terms are identified and TF-IDF value is computed for all sensible words. Each word is compared with all other meaningful word. The highest sensible of each word is computed. From the computed value semantic-graph is constructed. In graph construction, using Word Net, semantic relatedness between two terms is identified.

1.4 Requirements Traceability

Requirements traceability is an important activity undertaken as part of ensuring the quality of software in the early stages of the Software Development Life Cycle. Requirements tracing is a sub-area of requirements management within software engineering. It is mainly concerned with keeping track of the existence of requirements and providing bi-directional relationships between associated requirements [4]. The benefit for users is that the sub-area allows them to trace the origin of a requirement and keep track of every change made to it. The typical stages of requirements tracing of natural language documents are document parsing, candidate link generation, candidate link evaluation, and traceability analysis.

The first one, document parsing, addresses element extraction from both types of documents, use cases and requirements. Secondly, the candidate link generation performs a keyword matching on the keywords assigned to the requirements and use cases [11]. The candidate link evaluation assesses the previously generated links, in order to confirm that they are correct. At last, the traceability analysis deals with analyzing if there are use cases (lower-level) that satisfy a particular requirement (higher-level) [4]. In this work, we concentrate on adapting the Swarm Technique to the candidate link generation.

1.5 Terminology

The High and low level textual elements are called documents. Documents contain words or terms. The collection of all terms from all documents is called the dictionary or vocabulary. The collection of all terms in a document is called the document corpus. The inverted index lists all documents where a particular term occurs. Term frequency $TF_{t,d}$ is the count of how many times a particular term occurs in a document[4]. Inverse document frequency, IDF_t, is a calculated value

$$IDF_t = \log (N/ DF_t) \quad (1)$$

where, N is the total number of documents in a collection, and DF_t is document frequency i.e, the number of documents where a given term occurs in Eq.(1).

1.5.1 Measures

Tracing results are compared with an answer set of correct or ‘‘true’’ links. Results are then been evaluated using recall and precision. Precision and Recall are two standard measures used to

evaluate the algorithm's effectiveness. In addition to these measures, two other measures also called as Secondary measures namely DiffArr and MAP are applied for evaluation.

Precision P is calculated as the number of collection of document by number of relevant retrieved documents is shown in Eq. (2).

Precision, P = (No of Relevant Retrieved) / (No of Relevant in Collection) (2)

Recall R is calculated as the number of relevant retrieved document by number of retrieved documents is shown in Eq. (3).

Recall, R = (No of Relevant Retrieved) / (No of Retrieved) (3)

Using Eq. (2) and (3) F-measure is calculated as

F-Measure = ((Threshold) ^{2} + 1) P * R) / (Threshold) ^{2} P + R) (4)

DiffArr is calculated as the difference between the average similarity of the True Positives and False Positives is shown in Eq. (5) and MAP is calculated as the mean precision divided by relevant documents is shown in Eq. (6) where,

DiffArr = $\sum$ (True Positives) – $\sum$ (False Positives) (5)

MAP = ($\sum$Precision) / Relevant Documents) (6)

2. LITERATURE SURVEY

Jane Huffman Hayes, Wei-Keat kong's research focuses on a technique for requirements tracing, using Swarm Intelligence [4]. The applicability of Swarm Intelligence to the requirements tracing problem using pheromone communication and the common text around linking terms or words in order to find related textual documents are focused. In a nutshell, the technique can generate requirements traceability matrices (RTMs) between textual requirements artifacts (high level requirements traced to low level requirements, for example) with equivalent or better accuracy than traditional information retrieval techniques.

Hayes J, Dekhtyar A, Sundaram S, Howard [5] issues related to improving the overall quality of the requirements tracing process for Independent Verification and Validation analysts are addressed. It defines requirements for a tracing tool based on analyst responsibilities in the tracing process; and introduce several new measures for validating that the requirements have been satisfied; and present a prototype tool that built, RETRO (REquirements TRacing On-target), to address these requirements. The results of a study used to assess RETROýs support of requirements and requirement elements that can be measured objectively.

Diaz-Aviles E, Nejdl W, Schmidt-Thieme L (2009) "Swarming to rank for information retrieval"[2] *is a*n approach to automatically optimize the retrieval quality of ranking functions is explained. Taking a Swarm Intelligence perspective, the method *Swarm-Rank,* which is well-founded in a Particle Swarm Optimization framework. Swarm Rank learns a ranking function by optimizing the combination of various types of evidences such content and hyperlink features, while directly maximizing Mean Average Precision, a widely used evaluation measure in Information Retrieval.

Sundaram S, Hayes JH, Dekhtyar A, Holbrook A[10] states the generation of traceability links or traceability matrices is vital to many software engineering activities. It is also person-power intensive, time-consuming, error-prone, and lacks tool support. The activities that require traceability information include, but are not limited to, risk analysis, impact analysis, criticality assessment, test coverage analysis, and verification and validation of software systems. Information Retrieval (IR)[4] techniques have been shown to assist with the automated generation of traceability links by reducing the time it takes to generate the traceability mapping.

George Tsatsaronis, Irakilis Varlamis, Kjetil Nervag "SemanticRank: Ranking Keywords and Sentenses Using Semantic Graphs" [3] states that the selection of the most descriptive terms or passages from text is crucial for several tasks, such as feature extraction and summarization. Ranking is usually performed using statistical information from text (i.e.., frequency of co-occurrence, inverse document frequency, co-occurrence information). It states that *SemanticRank*, a graph-based ranking algorithm for keyword and sentence extraction from the text which constructs a semantic graph using implicit links, which are based on semantic relatedness between text nodes and consequently ranks nodes using different ranking algorithms.

3. METHODOLOGY

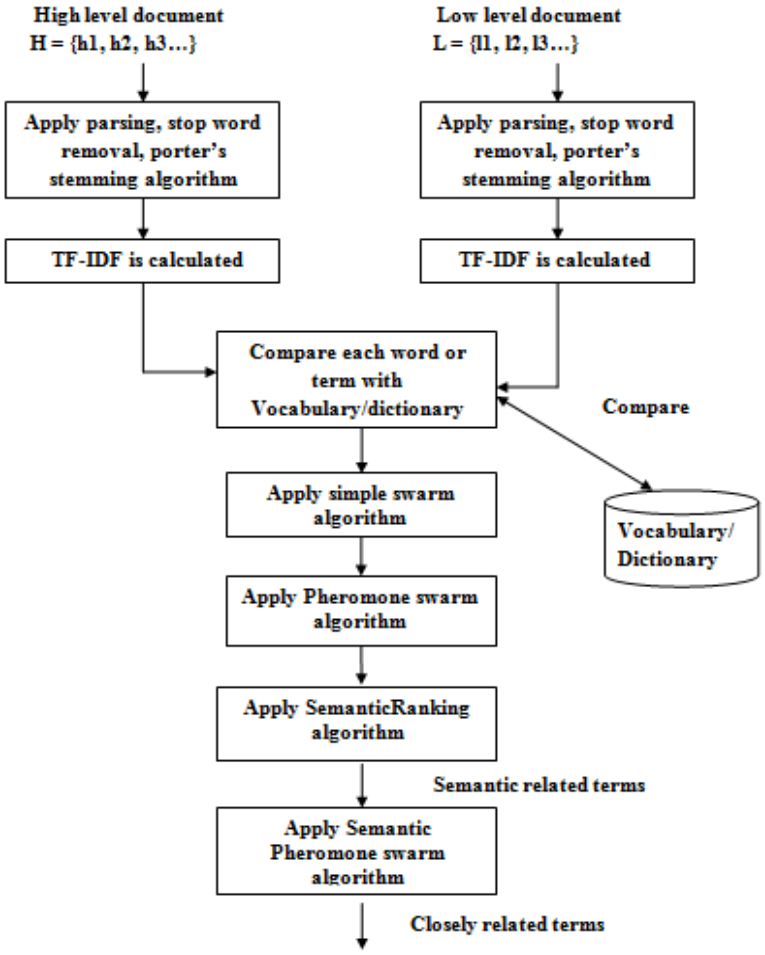

Figure 2. Overall design of candidate link generation.

A swarm agent starts from a high level textual document and follows a word or term that is present in the high level document via the common vocabulary. In the proposed system a simplified Ant colony algorithm, Simple and Pheromone Swarm are used to rank retrieved low-level requirement elements that may be relevant to high-level requirements [4]. Fig.2 describes that the documents are parsed, stop word such as (the, is, of, if, then) are removed, stemming (words are reduced to their stem such as 'comput-'For 'computer' and 'computing') is preformed [8]. Finally stemmed tokens are gathered, term frequency $TF_{t,d}$ and inverse document frequency IDF_t are calculated and TF-IDF value for each term should be found [4]. The TF-IDF values are sorted and the simple swarm selects the term randomly and count of terms that match the low level term from high level are calculated and sorted. The swarm concentrate on the top terms in the document rather than exploring all. Finally stemmed tokens are gathered. Term frequencies for each document and document frequencies for each term in the vocabulary are calculated. Term frequency $TF_{t,d}$ is the count of number of times a particular word / term occurred in a document. Inverse document frequency IDF_t is calculated using Eq. (1). Using $TF_{t,d}$, IDF_t, TF-IDF weight for each term is calculated using

$$\text{TF-IDF}_{t,d} = \text{TF}_{t,d} * \text{IDF}_t. \quad (7)$$

The swarm agents are given freedom to operate on their own, determining the search path based on the environment. The agent randomly selects a term or word. The count of terms that match the low level term from high level are calculated and sorted. Semantic relatedness between two terms can be computed based on highest value path connecting any pair of the terms. In finding highest value, the different meanings (senses) that appear between each word are determined. Pheromone swarm deposits on the links and terms to influence the path selection behaviour of a swarm agent. The highest value terms obtained from the semantic relatedness and TF-IDF are given as marking for Pheromone swarm. The marking deposited allows the agent to search, discover & guide swarm members to a target location.

4. SYSTEM DESIGN

4.1 Text Pre-Processing

High level documents are collected from customers and low level documents are collected from business analysis team [4]. For both high/low level document parsing is performed which results in tokens of words. From tokens, stop words are removed and stemming is performed using Porter's algorithm [8].

Porter's algorithm consists of following steps:

- Deals with plurals and past participles.
  E.g.: Motoring - Motor
- Deals with pattern matching on some common suffixes.
  E.g.: Happy - Happi,
  Relation - Relate.
- Deals with special word endings.
  E.g.: Hopeful - Hope.
- Checks the stripped word against more suffixes in case the word is compounded.
  E.g. Allowance - Allow,
  Inference - Infer.
- Check if the stripped word ends in a vowel and fixes it appropriately.
  E.g. Controll - control.

Finally stemmed tokens are gathered. Term frequencies for each document and document frequencies for each term in the vocabulary are calculated. Term frequency $TF_{t,d}$ is the count of number of times a particular word / term occurred in a document. Inverse document frequency IDF_t is calculated using Eq. (1). Using Eq. (7) $TF_{t,d}$, IDF_t, TF-IDF weight for each term is calculated.

4.2 Simple Swarm

High level and low level documents are given as input. High level document h terms are taken and terms are sorted based on TF-IDF weight [4]. The swarm agents are given freedom to operate on their own, determining the search path based on the environment. The agent randomly selects a term or word. A record of inverted index is maintained to list the occurrence of low level document. Then a Vocabulary set of all terms in all documents are maintained and link to low level documents are noted. Term frequency of terms is sorted. The count of terms that match the low level term from high level are calculated and sorted.

A loop through all high level elements is then executed that undertakes the following:

- A swarm agent is assigned to each high level element;
- The terms in that high level element are ordered by TF-IDF weight;
- The agent randomly selects a term from the top 10 or less terms (per the corpus for the whole document);
- using the selected term, the agent “crawls” from the high level element to the selected term in the inverted dictionary, i.e., the vocabulary space;
- once the agent descends to the level of the inverse dictionary, the agent again randomly selects from among the top 10 documents ordered by the term frequency;
-and once a low level element is picked, the agent moves down to that low level element [4].

When all agents reach the low level elements, we can determine candidate links.

4.3 Pheromone Swarm

Pheromone swarm deposits on the links and terms to influence the path selection behaviour of a swarm agent. The selection of the terms and links by the swarm agents is distinction between the simple swarm method and the swarm with pheromone method. There is no predetermined knowledge about the space being traversed. The agent’s decision on what term to select or what path to take is influenced by presence of pheromone markings on the inspected object, e.g., terms or link, the particular term is a neighbor to some other term in low-level document. Pheromone swarm deposited allows the agent to search, discover & guide swarm members to a target location.

A loop through all high level elements is then executed that undertakes the following:

- A swarm agent is assigned to each high level element;
- The terms in that high level element are ordered by TF-IDF weight value;
- the agent selects a term from the top 10 or less terms (per the corpus for the whole document) influenced by presence of pheromone markings on the inspected object ;
- using the selected term, the agent “crawls” from the high level element to the selected term in the inverted dictionary, i.e., the vocabulary space;

- once the agent descends to the level of the inverse dictionary, the agent again with the presence of pheromone markings selects from among the top 10 documents ordered by the term frequency; and once a low level element is picked, the agent moves down to that low level element[4].

When all agents reach the low level elements, we can determine candidate links.

4.4 Semantic Ranking

For the High Level document terms TF-IDF value is computed and semantic-graph is constructed. In graph construction, using Word Net, semantic relatedness between two terms is identified. Semantic relatedness between two terms can be computed based on highest value path connecting any pair of the terms. In finding highest value, the different meanings (senses) that appear between each word are determined [3]. The highest value terms obtained from the semantic relatedness are sorted in descending order. The documents with highest important word similarity are ranked as top position. This forms a Graph structure i.e., the document which get many No of important words gets the highest priority node. Other documents with least words are ranked next which is known as Page Rank.

4.5 Semantic Pheromone Swarm

Pheromone swarm deposits on the links and terms to influence the path selection behaviour of a swarm agent. The selection of the terms and links by the swarm agents is distinction between the simple swarm method and the swarm with pheromone method. There is no predetermined knowledge about the space being traversed. The agent's decision on what term to select or what path to take is influenced by presence of pheromone markings on the inspected object, e.g., terms or link, the particular term is a neighbor to some other term in low-level document. Pheromone swarm deposited allows the agent to search, discover & guide swarm members to a target location.

A loop through all high level elements is then executed that undertakes the following:

- A swarm agent is assigned to each high level element;
- The terms in that high level element are ordered by TF-IDF weight and semantic relatedness value;
- the agent selects a term from the top 10 or less terms (per the corpus for the whole document) influenced by presence of pheromone markings on the inspected object ;
- using the selected term, the agent "crawls" from the high level element to the selected term in the inverted dictionary, i.e., the vocabulary space;
- once the agent descends to the level of the inverse dictionary, the agent again with the presence of pheromone markings selects from among the top 10 documents ordered by the term frequency; and once a low level element is picked, the agent moves down to that low level element[4].

When all agents reach the low level elements, we can determine candidate links.

5. EXPERIMENTAL RESULTS

The Fig 3 shows the high level and low level documents are taken, text pre-processing such as parsing, stop word removal, stemming are preformed. Finally stemmed tokens are gathered.

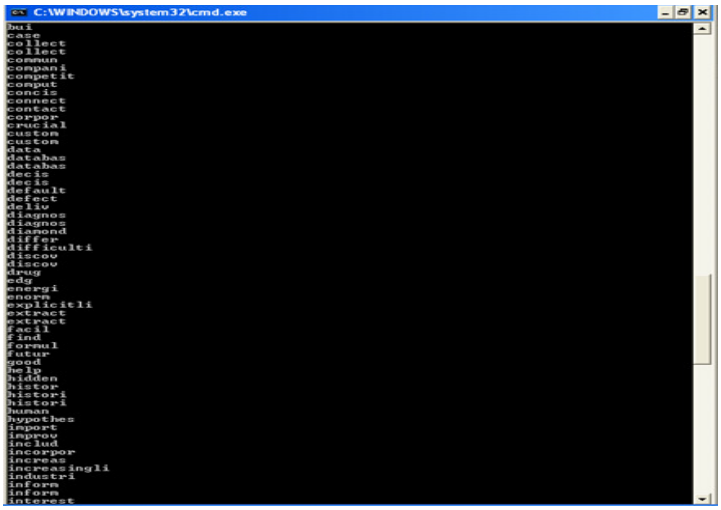

Figure 3. Stemmed output

Fig.4 shows for the gathered words term frequency $TF_{t,d}$ and inverse document frequency IDF_t are calculated and TF-IDF value for each term are found. The TF-IDF values are sorted and fig.5 shows the TF-IDF values are sorted and the simple swarm selects the term randomly and count of terms that match the low level term from high level are calculated and sorted.

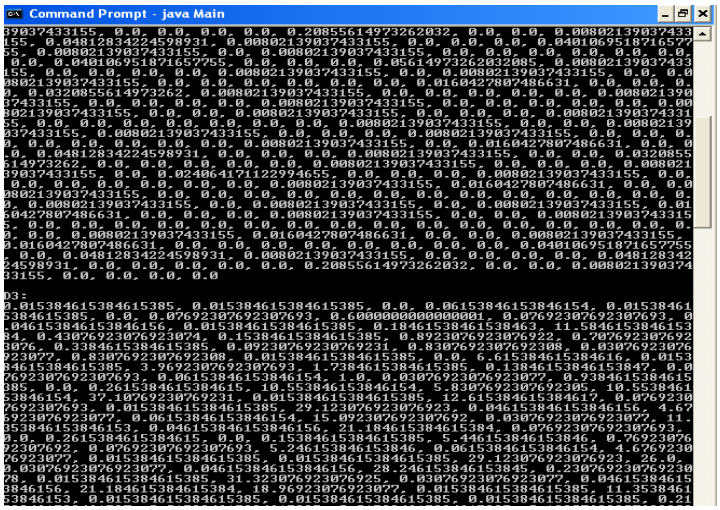

Figure 4. TF-IDF Values

Fig.6 shows the TF-IDF values are sorted and the Pheromone swarm selects the term comparing the neighborhood terms, its TF-IDF values and count of terms that match the low level term from high level are calculated and sorted. Fig.7 shows that using similarity and TF-IDF values semantic graph is constructed and number of times the term's occurrences in the document is known as page rank and the terms are sorted. Fig.8 shows that semantic pheromone swarm selects the term comparing the neighborhood terms, its TF-IDF and semantic relatedness values and count of terms that match the low level term from high level are calculated and sorted.

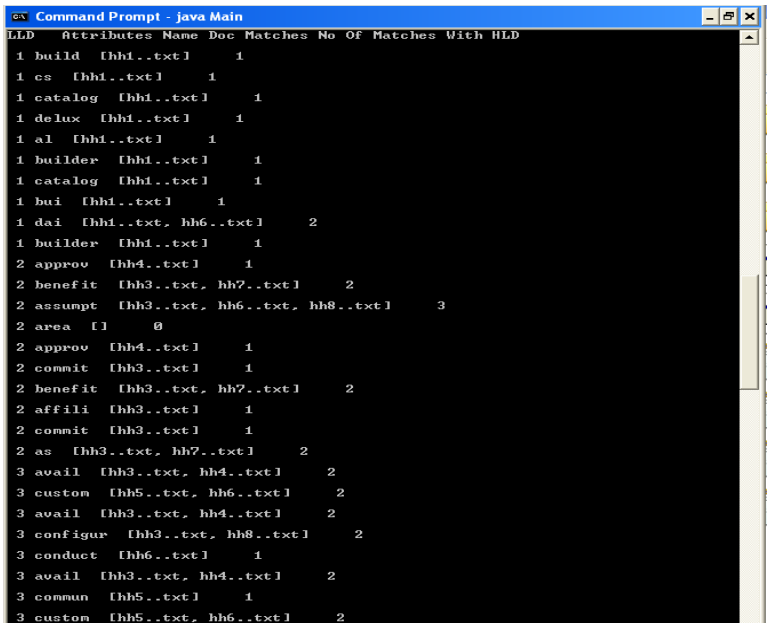

Figure 5. Implementation of Simple Swarm

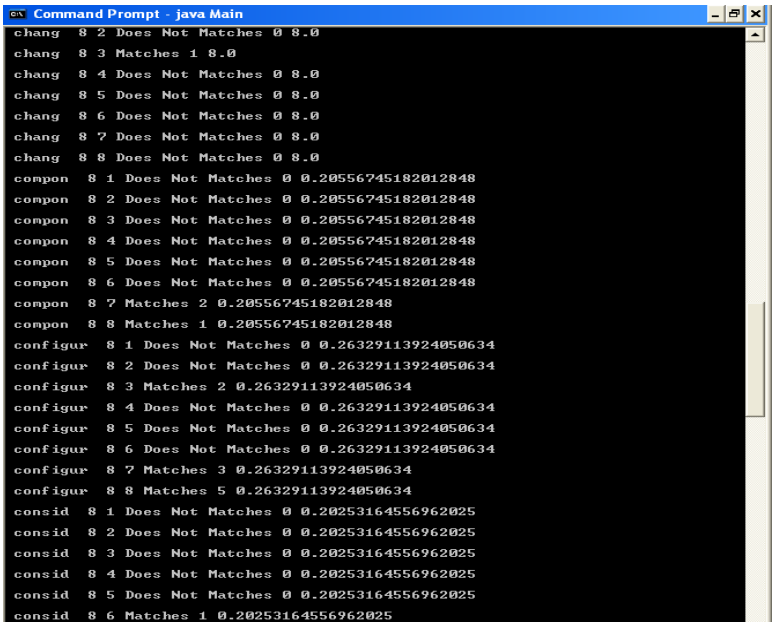

Figure 6. Implementation of Pheromone Swarm

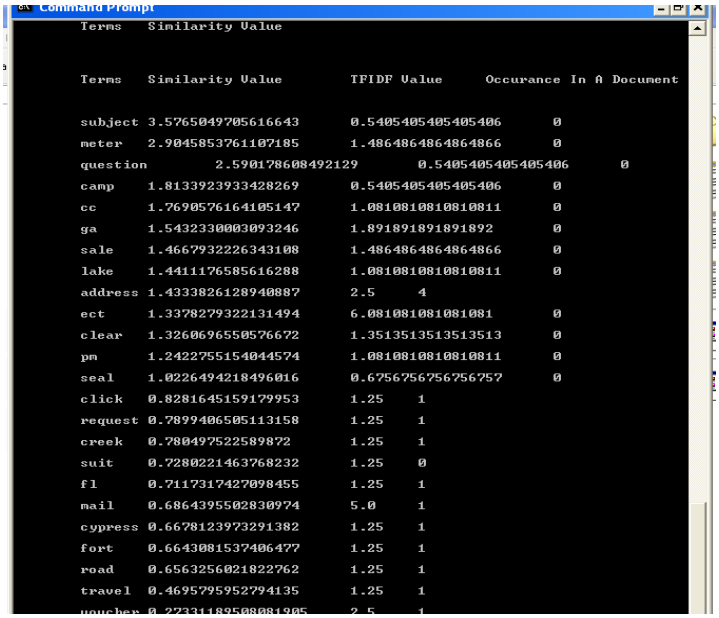

Figure 7. Implementation of Semantic Graph

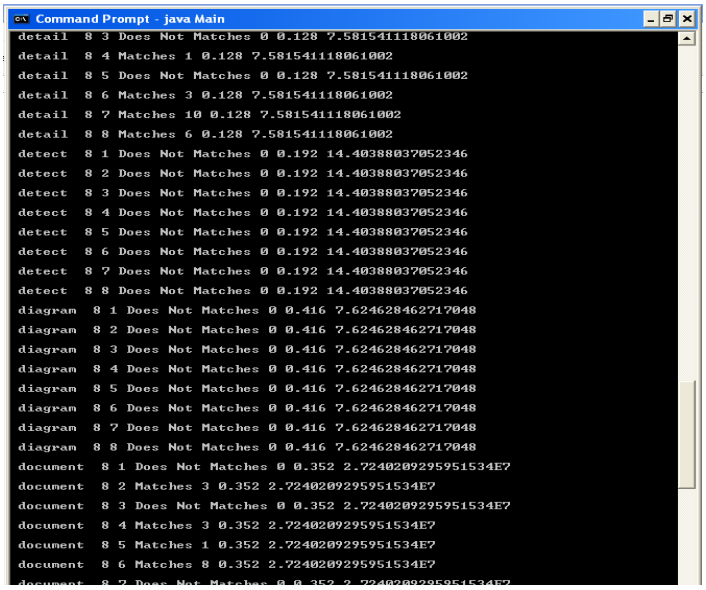

Figure 8. Implementation of Semantic Pheromone Swarm

In this project, techniques such as Simple Swarm, Pheromone Swarm and Semantic ranking algorithm are proposed. For HLD, LLD documents Simple and Pheromone Swarm techniques are implemented and Candidate link is generated between the documents. In addition to this, Semantic Ranking algorithm is implemented along with Pheromone Swarm and results are obtained. Among these algorithms Semantic Pheromone Swarm provide better result of

Candidate Link between the documents. Precision, Recall, F-Measure, DiffArr and MAP values are calculated separately for evaluation between Simple, Pheromone and Semantic Pheromone Swarm algorithms.

Table I
Results of Simple Swarm

Threshold	Precision	Recall	F-Measure	Map	DiffArr
0.1	0.718	0.143	0.311	0.718	0.188
0.2	0.182	0.365	0.147	0.450	0.224
0.3	0.270	0.541	0.130	0.390	0.226
0.4	0.370	0.741	0.123	0.385	0.231
0.5	0.455	0.91	0.114	0.399	0.232
0.6	0.560	0.112	0.284	0.426	0.237
0.7	0.638	0.127	0.272	0.456	0.233
0.8	0.721	0.144	0.268	0.489	0.232
0.9	0.815	0.163	0.271	0.525	0.231
1	0.905	0.181	0.275	0.563	0.234

Table II
Results of Pheromone Swarm

Threshold	Precision	Recall	F-Measure	Map	DiffArr
0.1	0.0125	0.025	0.02487562	0.0125	0.209091
0.2	0.025	0.05	0.04901961	0.01875	0.213318
0.3	0.16875	0.3375	0.32296651	0.06875	0.158816
0.4	0.05	0.1	0.09259259	0.064063	0.125901
0.5	0.0625	0.125	0.11111111	0.06375	0.275041
0.6	0.075	0.15	0.12711864	0.065625	0.275041
0.7	0.0875	0.175	0.14056225	0.06875	0.117526
0.8	0.0999	0.199	0.15151515	0.072656	0.148589
0.9	0.1125	0.2249	0.16014235	0.077083	0.319882
1	0.0625	0.1249	0.08333333	0.075625	0.319882

Table III
Results of Semantic Pheromone Swarm

Threshold	Precision	Recall	F-Measure	Map	DiffArr
0.1	0.225	0.35	0.225	0.225	0.093
0.2	0.325	0.45	0.328	0.275	0.134
0.3	0.8625	1	0.872	0.470	0.204
0.4	0.525	0.65	0.539	0.484	0.171
0.5	0.625	0.75	0.646	0.512	0.218
0.6	0.725	0.85	0.754	0.547	0.217
0.7	0.825	0.95	0.862	0.587	0.109
0.8	0.924	1	0.952	0.629	0.142
0.9	1	1	1	0.670	1
1	1	1	1	0.703	1

Table 6.1 represents Precision, Recall, F-Measure, Map, and DiffArr of Simple Swarm for SRS documents. Table 6.2 provides Precision, Recall, F-Measure, Map, and DiffArr of Pheromone Swarm for SRS documents. Table 6.3 provides Precision, Recall, F-Measure, Map, and DiffArr of Semantic Pheromone Swarm for SRS documents. By comparing Semantic Pheromone swarm with Simple Swarm and Pheromone swarm results reveal that Semantic Pheromone Swarm gives better results for threshold values like 0.89,0.99. Using Semantic Pheromone Swarm Candidate Link is generated which is shown with high DiffArr and MAP values.

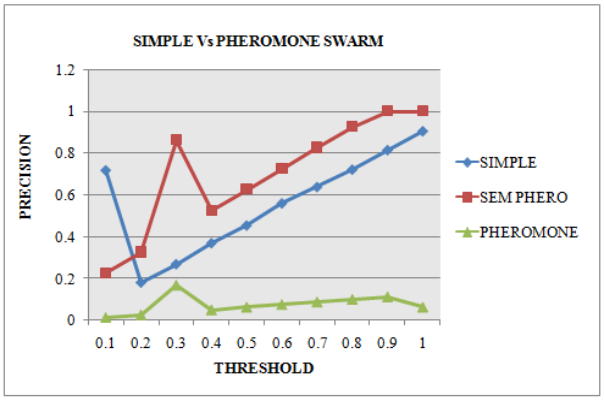

Figure 8. Precision Vs Threshold

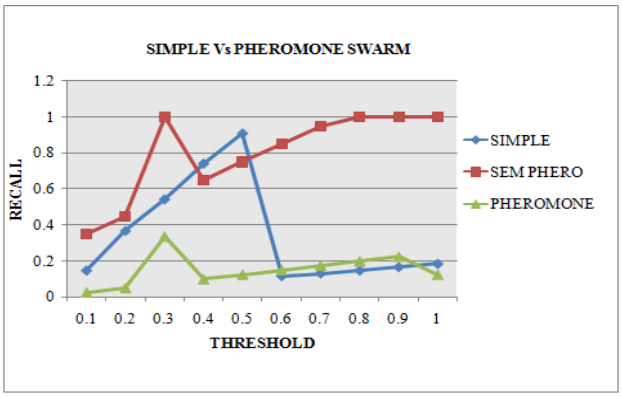

Figure 9. Recall Vs Threshold

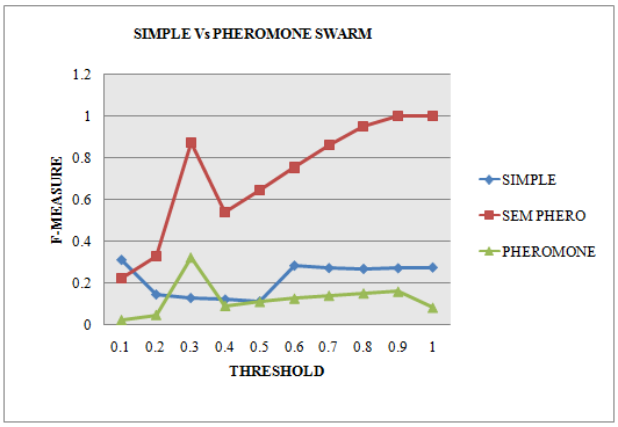

Figure 10. F-Measure Vs Threshold

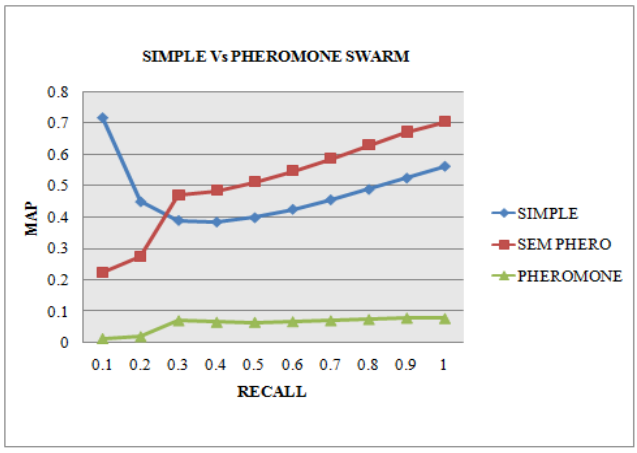

Figure 11. Map Vs Recall

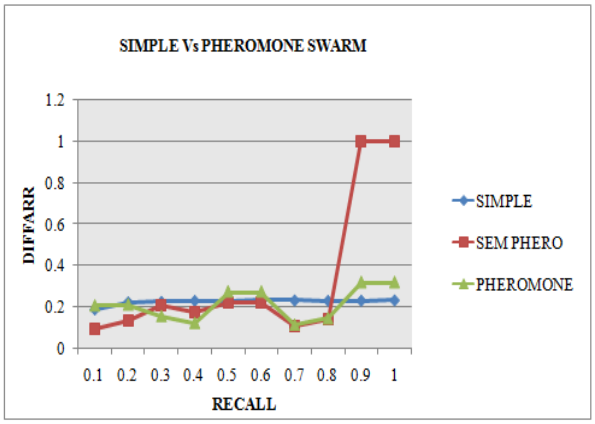

Figure 12. DiffArr Vs Recall

Precision, Recall, F-Measure of Simple, Pheromone and Semantic Pheromone Swarm are compared with Threshold values as shown in Fig. 8, Fig. 9, and Fig. 10. Map and DiffArr of Simple, Pheromone and Semantic Pheromone Swarm are compared with Recall values of Semantic Pheromone Swarm are shown in Fig. 11 and Fig. 12. Here high DiffArr and high MAP value of Semantic Pheromone Swarm implies that Candidate link is generated more effectively using Semantic Pheromone Swarm technique when compared with Simple and Pheromone Swarm [4]. High DiffArr refers that the average relevance of true positives is higher than the false positives and high MAP implies precision is high at various recall values [8].

6. CONCLUSION & FUTURE WORK

The high level and low level documents are taken, text pre-processing such as parsing, stop word removal, stemming is preformed. Finally stemmed tokens are gathered, term frequency $TF_{t,d}$ and inverse document frequency IDF_t are calculated and TF-IDF value for each term are found. The TF-IDF values are sorted and the simple swarm selects the term randomly and count of terms that match the low level term from high level are calculated and sorted. The swarm concentrate on the top terms in the document rather than exploring all. Semantic ranking algorithm has been applied and semantic graph has been obtained. For the semantic graph pheromone algorithm has been applied and list of agent count has been obtained. Thus for the low level documents and high level documents, candidate link is generated.

As a future work, it can be expanded by using a thesaurus. Latent Semantic Analysis [2] technique is able to match similarities between multiword expressions, abbreviations, and alpha-numeric phrases [3] permitting the agents to discover links not only through a single term but through term synonyms.

REFERENCES

[1] G.Antoniol, G.Canfora, G.casazza, A.D.Lucia, and E.Merlo, “Recovering Traceability Links between Code and Documentation,” IEEE Trans.Softw.Eng., vol.28, 2002, pp970-983.

[2] Diaz-Aviles E, Nejdl W, Schmidt-Thieme L (2009) “Swarming to rank for information retrieval”. *Proceedings of the 11th annual conference on genetic and evolutionary computation*, pp 9–16.

[3] George Tsatsaronis, Irakilis Varlamis, Kjetil Nervag "SemanticRank: Ranking Keywords and Sentenses Using Semantic Graphs" *Proceedings of the 23rd international conference* on computational linguistics (coling 2010),pp 1074-1082.
[4] Hakim sultanov, Jane Huffman Hayes, Wei-Keat kong "Application of Swarm Techniques to requirement tracing" *Springkerlink* May 2011.
[5] Hayes J, Dekhtyar A, Sundaram S, Howard (2004) "Helping analysts trace requirements: an objective look". *Proceedings of the 12th IEEE international conference in requirements engineering*, *2004*, pp 249–259.
[6] Hayes JH, Dekhtyar A, Osborne J (2003) "Improving requirements tracing via information retrieval". *Proceedings of the 11th IEEE international conference on* requirement engineering, p 138.
[7] J.H. Hayes, A. Dekhtyar, S. Sundaram, and S.Howard, "Helping Analysts Trace Requirements: An Objective Look," 2004.
[8] M. Porter, "An algorithm for suffix stripping," *program*, vol. 14, 1980, pp. 130-137.
[9] Sultanov H, Hayes JH(2010) "Application of Swarm Techniques to requirement engineering: requirements tracing". *Proceedings of the 18th international* requirement engineering *conference*, Sydney.
[10] Sundaram S, Hayes JH, Dekhtyar A, Holbrook A (2010) "Assessing traceability of software engineering artifacts". *Require Eng* J 15(3):211-220.
[11] Xuchang Zou, R. Settimi, and J. Cleland-Huang, "Phrasing in Dynamic Requirements Trace Retrieva," 2006, pp. 265-272.

Authors

Ms. Susan Geethu.D.K was born in Coimbatore, India in 1988. She received Bachelor of Technology degree in Information Technology under Anna University, Chennai in 2009. She is currently pursuing Master of Engineering degree in Computer Science and Engineering under Anna University, Chennai, India.

Ms. R.Subha received B.E in Computer Science and Engineering from Periyar University and M.E in Software Engineering from Anna University, Chennai in 2002 and 2006 respectively. At Present, she is working as Assistant Professor in the department of Computer Science & Engg, Sri Krishna College of Technology, Coimbatore. She is currently pursuing Ph.D under Anna University, Coimbatore. Her research interest includes Software Engineering.