ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 4, Issue 8 , August 2017

Searching Comparatively Better Result From Agglomerative algorithm

[1] Dipalee Prakash More, [2] Prof. Ujawala M Patil
[1]PG Student, [2] Associate Professor

Abstract— **In this world the Internet has become very casual for searching, the user appears to use it every time, even they need to search keyword from any information query, search relevant word and a lot more. Also, people use search engine like Google, Bing when they are willing to search something, wants to use some relevant information or go to their synonyms. But searching for correct result requires more time and less execution speed even they produce multiple choices. So, this process is very confusing for users to decide one correct keyword amid the many results as a seek engine show overall results. For these reasons, the present paper centering on generally showing the final result and to show exact keyword. Intended to the agglomerative algorithmic approach is used which aim to generate exact keyword in less time and reducing computational cost. The agglomerative approach is very useful for knowing the best result from requiring query candidate.**

Keywords— **Agglomerative Algorithm, Anchor-Based Pruning Solution, Baseline Solution.**

INTRODUCTION

Searching is one of the best ways to know the information content from structured and semi-structured data, but the user having the knowledge of sophisticated query language [1]. In a process of information retrieval, the node chooses to detect a list of relevant documents For example, when someone is interested to search a particular information purchasing product. The information like Name of person, Mobile Number of people, City of person, Qualification of Pearson for details otherwise in harvests similar AC, Refrigerator, bike for buying, he/she would famine near recognize the other prospect earlier accomplishment the absolute verdict. Trendy this domain, persons effortlessly search further stuffs however solitary once they recognize all around individuals articles. For specimen, Searching information about an AC or refrigerator for going out to the market is tranquil as we recognize a slight tad around these gadgets. That wealth searching receipts in a higher data. Customer Information, the core information chart is the cradles that provide such a high knowledge of product and try to satisfy the need of searching. As most of the folks use seek devices to search keyword and number of keywords are produce hence it require more time. So we presented method centers solitary on display the superlative option and best unconventional which is habitually vital to sort the ending resolution. The user's contains query which is easier to search intention with query can be identified, a user mutual action may require more time when dataset size is large. To solve this, problem, in existing paper develops a method of diversifying query suggestions to user's based on result to be generated. At the time of performing the users may choose to adapt their original queries based on their return diversify result of query suggestion. For example, Consider query q= {creation, day} over the DBLP dataset. There are 9,596 documents containing the keyword "creation", and 6,587 documents containing keyword "day", which contributes1,027 results that contains the two given keyword together. Then directly search keyword, the process become time consuming, not user friendly because number of results will be generated. At the time of more number of results occurs, then use, competent systems [3], [4], it removes an uncertain and frequent occurred resulting.

Diversification is one of the methods uses to find the exact search keyword. In the diversification process, firstly deduces the mutually related feature terms of gauging its consequence to unique result then proposed result set. In Existing paper, we produce a number of results from input query keyword. It is very confusing to decide which is useful; hence this method is used to prepare better result.

II. REVIEW

The searching method before is done with structured and semi-structured data has implemented by using the information retrieved. But specifically detecting search keyword was not done yet.

Y. Chen et al. Presented a novel approach to keyword search for structure and semi-structured data, search result generates and improving the search result, by using information integration and analysis. It provides a lightweight method of integration as a database selection by keyword relationship graph, query generation, analytical processing. Also describe the future research model as diverse data model, improve quality for search result and evaluation [1].

L. Guo et al. Present a method of XML dataset. It describes the problem on efficiently occurred search result using the model structure of the database and semantic of query. Specifically, describe the XML data, the result of a query and ranking method. It also describes the index structures and query evaluation techniques [2].
C. Sun et al. Present a novel approach of keyword searches on smallest, lowest common ancestor from an XML document by using their anchor node and their properties, also provide incremental multiway common ancestor and at the last analyze this process [3].

Y. Xu et al. Present a novel approach to efficient keyword search, also present the search result on rooting the sub tree. The index lookup eager algorithm for indexing purpose and also uses scan eager algorithm for the scan node list, stack algorithm for merges all keyword lists nodes [4].

The work on keyword search is associated with information retrieval and re-ranking query interpretations in information extraction. Specifically, the most appropriate work is by Clarkel and E. Demidove [5, 6] on real-world datasets demonstrates that search results. Their methods applied redundancy and novelty [5] learned from relevance estimation. In their experimental methods typically can attain normalize discount cumulative gain set up as the standard evaluation method.

III. MRTHODOLOGY

In general, searching the relevant information and objective methods like generation procedure. Optimization seeks values of variables that lead to an optimal value of the function that is to be optimized in Figure 1.

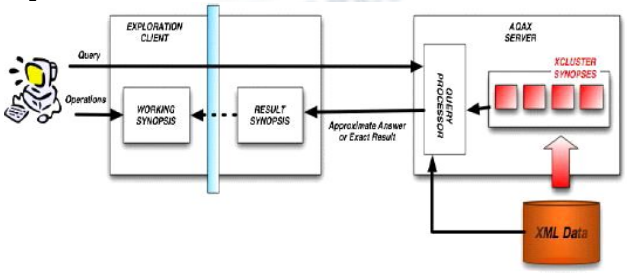


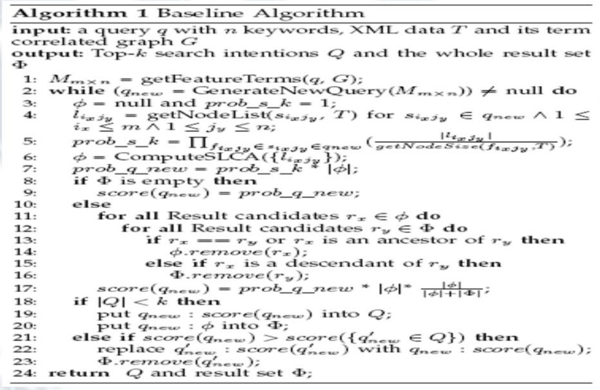
Figure1. System Structure.

A. Baseline Solution

The baseline solution, retrieve the relevant terms with their score. Then, list out the query candidates. The mutual information score is used to search their keyword result.

- Steps of Baseline Solution

1. Construct matrix Mm*n by using pre-computed relevant feature terms of graph G.
2. Then generate new query qnew from matrix Mm*n by function GenerateNewQuery ().
3. Then write new query candidate qnew in order of descending, according to their score.
4. Compute the common ancestor result of qnew by retrieving the node list of keyword feature terms.
5. Then work out the probable result of causing witnessed query q.
6. Then compute the common ancestor result of actual result and old result, in order to obtain diversified common ancestor result.
7. And at the last, compare new result with old result and interchange outright ones in Q.



```
Algorithm 1 Baseline Algorithm
input: a query $q$ with $n$ keywords, XML data $T$ and its term correlated graph $G$
output: Top-$k$ search intentions $Q$ and the whole result set $\Phi$
$M_{m \times n}$ = getFeatureTerms($q$, $G$);
while ($q_{new}$ = GenerateNewQuery($M_{m \times n}$)) $\neq$ null do
    $\phi$ = null and $prob\_s\_k = 1$;
    $l_{i_x j_y}$ = getNodeList($s_{i_x j_y}$, $T$) for $s_{i_x j_y} \in q_{new} \wedge 1 \leq i_x \leq m \wedge 1 \leq j_y \leq n$;
    $prob\_s\_k = \prod_{f_{i_x j_y} \in s_{i_x j_y} \in q_{new}} (\frac{|l_{i_x j_y}|}{getNodeSize(f_{i_x j_y}, T)})$;
    $\phi$ = ComputeSLCA($\{l_{i_x j_y}\}$);
    $prob\_q\_new = prob\_s\_k * |\phi|$;
    if $\Phi$ is empty then
        $score(q_{new}) = prob\_q\_new$;
    else
        for all Result candidates $r_x \in \phi$ do
            for all Result candidates $r_y \in \Phi$ do
                if $r_x == r_y$ or $r_x$ is an ancestor of $r_y$ then
                    $\phi.remove(r_x)$;
                else if $r_x$ is a descendant of $r_y$ then
                    $\Phi.remove(r_y)$;
        $score(q_{new}) = prob\_q\_new * |\phi| * \frac{|\phi|}{|\phi|+|\Phi|}$;
    if $|Q| < k$ then
        put $q_{new} : score(q_{new})$ into $Q$;
        put $q_{new} : \phi$ into $\Phi$;
    else if $score(q_{new}) > score(\{q'_{new} \in Q\})$ then
        replace $q'_{new} : score(q'_{new})$ with $q_{new} : score(q_{new})$;
        $\Phi.remove(q'_{new})$;
return $Q$ and result set $\Phi$;
```

Figure2. Baseline Algorithm.

B. Anchor-Based Pruning Solution

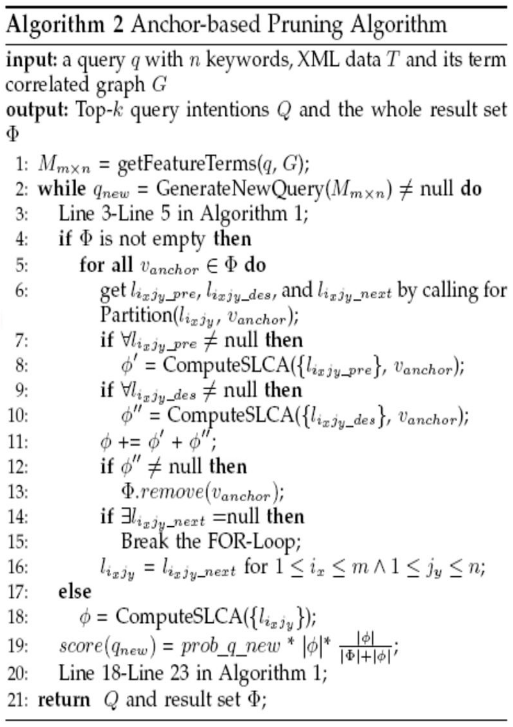
By studying the previous method of search result, we can compute the final one execution rate of this resolution done on the determining ancestor result and destroying unnecessary ancestor result from a fresh and old produced outcome set.
The anchor-based pruning, solution design to escape gratuitous execution costs by analyzing their interrelationship between intermediate ancestor result.

- ***Steps of Anchor–Based Pruning Solution***

1. Firstly constructs matrix Mm*n of feature terms, retrieve the list of nodes by maintaining their index term.
2. And above, then evaluate q related when the query is qnew.
3. Then apply the intermediate common ancestor results of old result as node efficiently computes

new ancestor result.

4. Caused query, we can find the common ancestor consequences expending previously implemented searching method as a baseline solution.
5. The prune result of ancestor is considered as a first query and the list of nodes of the second queries for reducing its cost of evaluation.
6. Anchor node for every list of nodes of keyword in present fresh query, we grow many active grades of nodes using index term by using the Partition () method.
7. The common ancestor results are different from the old ancestor node, then they will be canned as new distinguishable result and old ancestor result will remove from the implementing result set.
8. At the last, we record the score and result of the new query.



Algorithm 2 Anchor-based Pruning Algorithm

input: a query q with n keywords, XML data T and its term correlated graph G
output: Top-k query intentions Q and the whole result set Φ

```
1: $M_{m\times n}$ = getFeatureTerms($q$, $G$);
2: while $q_{new}$ = GenerateNewQuery($M_{m\times n}$) $\neq$ null do
3:   Line 3-Line 5 in Algorithm 1;
4:   if $\Phi$ is not empty then
5:     for all $v_{anchor} \in \Phi$ do
6:       get $l_{i_x j_y\_pre}$, $l_{i_x j_y\_des}$, and $l_{i_x j_y\_next}$ by calling for
         Partition($l_{i_x j_y}$, $v_{anchor}$);
7:       if $\forall l_{i_x j_y\_pre} \neq$ null then
8:         $\phi'$ = ComputeSLCA($\{l_{i_x j_y\_pre}\}$, $v_{anchor}$);
9:       if $\forall l_{i_x j_y\_des} \neq$ null then
10:        $\phi''$ = ComputeSLCA($\{l_{i_x j_y\_des}\}$, $v_{anchor}$);
11:      $\phi$ += $\phi' + \phi''$;
12:      if $\phi'' \neq$ null then
13:        $\Phi.remove(v_{anchor})$;
14:      if $\exists l_{i_x j_y\_next}$ =null then
15:        Break the FOR-Loop;
16:      $l_{i_x j_y} = l_{i_x j_y\_next}$ for $1 \leq i_x \leq m \wedge 1 \leq j_y \leq n$;
17:  else
18:    $\phi$ = ComputeSLCA($\{l_{i_x j_y}\}$);
19:  $score(q_{new}) = prob\_q\_new * |\phi| * \frac{|\phi|}{|\Phi|+|\phi|}$;
20:  Line 18-Line 23 in Algorithm 1;
21: return $Q$ and result set $\Phi$;
```

Figure3. Anchor-Based Pruning Algorithm

C. Agglomerative Algorithm.

By analyzing the base paper method, the user involution is useful to fine search value is of keyword queries, a user's searching process may require more time when the size of the pertinent result set is large. In our presented paper use the agglomerative algorithm works by number of values compounded in unique result, their distance between the data point which required less time and more execution speed. In existing paper implementation, the cluster is divided in n number of document with same size [8]. Then, it sets centroid as a maximum number of keyword present in that document, these document values consider as a centroid value. Every document file having one relevance value [9]. After that, centroid matches to those documents that value is similar. And only analyze that document for example, it has total 10 documents Centroid calculates as a maximum number of documents having common value. So, 4 documents having common value as a 5. Hence, only 4 document matches there centroid so remaining 6 not to analyze their value, so the execution process will not carry out and time require to process execution is also less and speed also increases [10].

• ***Steps Of Agglomerative Algorithm***

1. Start.
2. K (0) =0 and r a categorization value initially is r=0.
3. Finding lowest distance in running clustering as a pair of (r), (s) allowing to d [(r) (s)] = mn d [(i)(j)] when bottom to all terminated.
4. r=r+1
5. Join (r), (s) into single cluster from another cluster m.
6. K (r) = d [(r)(s)].
7. Inform reserve value and adding to form new cluster, as a (r, s).
8. Stop.

IV. EXPERIMENTAL RESULTS

A. Data Source

To perform the experiments of base paper, the data sets are requested from DBLP [11] this is the real data set and XMark [12] is an artificial XML dataset for testing the diversification keyword result. The size of DBLP dataset is 227 MB, it has element 3332130, attribute is 404276, maximum depth is 6 and average depth is 2090228. The size of the artificial XMark dataset is 198 MB, having element 26859, attribute is 5689, maximum depth is 4 and the average depth is 2.668.

B. Analysis.

We analyze the existing paper method, Baseline solution is to repossess the pertinent piece values with the in elevation score. Then generate list occurred value

keyword that is sorted in descending order for the total score. And then, finally compute the common ancestor result for is querying candidate measure their score. Different from other search engine, existing system work need to appraise multiple query candidate and generate entire result set.
By analyzing anchor-based pruning, solution, limitation of baseline solution as more computational cost and time is overcome in the anchor-based solution. Anchor-based parallel sharing solution use by working the similarly by virtue of corresponding of diversification for search keyword and reduction in the perennial scanning of the similar list of nodes, which is a less time-consuming process and more execution speed..

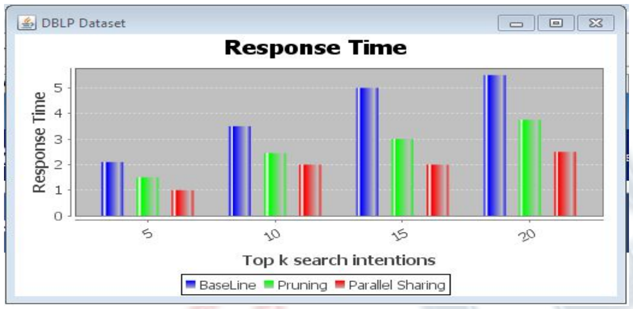


Figure 5. Average time cost of queries.

Fig 5 shows the average time cost of queries, using the baseline by using an information retrieval process and removing duplicate results. And anchor-based pruning, solution used by avoiding unqualified ancestor result. Parallel sharing used by partitioning in similar part. So, the result is improved by their method.

C. Our Improvement.

In the base paper there is a limitation in C. Sun et al. A method that, their method many numbers of query are evaluated using structure value mathods. For example, keyword Day generates result as age, epoch, term, period, cycle. But in our method we have overcome this limitation that means our method can generate exact key word in minimum time with high speed. We use the agglomerative algorithm to improve our final result. These methods use the step wise processing as firstly, Assign each object to a separate cluster, then Evaluate all pairwise distances between clusters.In our presented paper, we use an agglomerative algorithm for improving their results as reduce their response time by using the centroid identification method. In the centroid identification method, firstly computes centroid as a maximum result occurred document value. Then it matches only that document. Remaining document is not being analyzed. So, visiting document process is less hence time require also less and execution speed increases. Fig 4 shows the nDCG value for Google and Bing search engine. Fig 5 shows the Response time require to keyword search for Baseline, Anchor-based pruning and Parallel sharing solution. Fig 6 shows the improve results for keyword search by comparing base paper method as Baseline, anchor-based pruning and parallel sharing solution with our propose method as agglomerative algorithm.

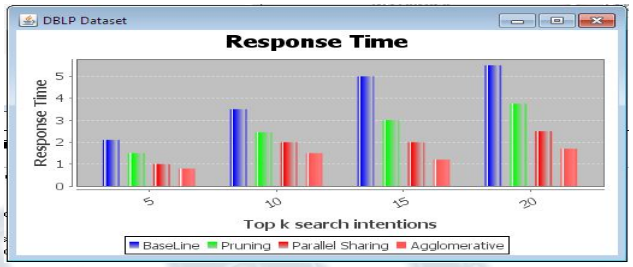


Figure 6. Comparison Result Between Base Paper and Our Improve Method Result.

CONCLUSION

Someone customs a search engine to know the better in input candidate query he or she will produce many numbers of results in the data value. Nonetheless baseline and anchor-based pruning, solution focuses only on query candidate means it directly gets input information is better as well as it also give number of choices. So, this baseline and anchor-based pruning, solution is very useful for effectively answer keyword queries. At baseline and anchor-based pruning, solution, the general users to search a vast amount of data is very difficult due to the ambiguity in keyword query and difficulty to effective answer to query candidate. Agglomerative algorithm is useful for searching best result by their centroid method in less time and also reduces the computational cost. It significantly improves the searching keyword from the dataset. This method can effectively use for input query search or information retrieval system. Also, these approaches only use qualified results and get it in a short time.

REFERENCES

[1] Y. Chen, W. Wang, Z. Liu, and X. Lin. 2009.Keyword search on structured and semi-structured data, in SIGMODConference, pp:b1005–1010.

[2] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram. 2003.Xrank: Ranked keyword search over xml documents, in SIGMOD Conference, pp:16–27.

[3] C. Sun, C. Y. Chan, and A. K. Goenka.2007.Multiwayslcabased keyword search in xml data, in WWW, pp:1043–1052.

[4] Y. Xu and Y. Papakonstantinou.2005.Efficient keyword search for smallest lcas in xml databases, in SIGMOD Conference, pp: 537–538.

[5] C. L. A. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova,A. Ashkan, S. Buttcher, and I. MacKinnon.2008.Novelty and diversity in information retrieval evaluation, in SIGIR, pp:659–666.

[6] E. Demidova, P. Fankhauser, X. Zhou, and W. Nejdl. 2010. DivQ:Diversification for keyword search over structured databases, inProc. SIGIR, pp:331–338.

[7] D. Mullner.2011.Modern Hierarchical Agglomerative clustering algorithm in arXiv:1109.2378vl.

[8] N. Slonim and N. Tishby, Agglomerative Aglorithm Bottlneck.

[9] K. Sasirekha and P. Baby. 2013.Agglomerative Hierarchical clustering algorithm: A Review, in IJSRP, vol. 3, Issue 3,pp:1-3.

[10] M. Hasan, A. Mueen, V. J. Tsotras, and E. J. Keogh. 2012. Diversifying query results on semi-structured data, in CIKM, pp:2099–2103.

[11] http://dblp.uni-trier.de/xml/.

12] http://monetdb.cwi.nl/xml/.

[13] A. Angel and N. Koudas. 2011. Efficient diversity-aware search, in SIGMOD Conference, pp: 781–79.