An in-depth study of the syntactic and semantic analysis of search graph matching algorithms, to the degree of sub graphs

Problem statement

Evaluating the isomorphic (syntactic) and the symmetric relationship between two or more di-graphs using search algorithms, and then comparing the similarities of the graphs is an expensive enterprise. This is because the procedures that are required to achieve this task are often expensive and the large data sets that are used for symmetric analysis and comparison often from different and unrelated sources. And thus cases where the degree of similarity between two graphs is evaluated, only to find that the graphs are not syntactically similar after a large amount of time and resources have been exhausted is very common, particular in cases where the graphs contain large datasets.

Search algorithms such as the Similarity flooding algorithm[1] and the VF2 algorithm[2] compare the syntactic and symmetric relationship between graphs by iterating through the graphs from one node to another, and evaluating once node at a time in order to perform similarity relation comparison, and because the comparison operation requires processing of all the nodes, the larger the graphs are the more time and resources must be used in order to completely evaluate this relation between the graphs. Thus better search algorithms and strategies are required in order to improve the efficiency of the comparison.

Objective

In order to improve on the quality of the comparison between di-graphs, efficient search algorithms that perform comprehensive comparison on the syntactical and symmetrical relationships between graphs as well as their sub-graphs are investigate in order to perform comprehensive graph matching relative the generic approach of simply iterative through each graph node and comparing its content and position. The graph search algorithms that are required and thus investigated must be efficient in terms of their respective space and time complexities.

Apart from being efficient, the algorithms must also perform graph matching on lower levels of granularity of the graphs, thus the algorithms must also compare the syntactic and semantic comparisons

on the sub-graphs for all the permutations of the two graphs sub-graphs, so that the optimal graph matched result can be generated.

Thus the algorithms must quantify their result and record the local maximums, and once the graph matching is complete they must provide the global maximum that will be used to evaluate the algorithms and determine the best amongst them.

Scope

The scope of the proposed research deals with measuring the degree of how much two graphs are equal to each other, using graph search algorithms such as the VF2 algorithm [2].This is accomplished by matching the two graphs as well as their sub-graphs.

This measure evaluates the best match, as well as the efficiency of the algorithms in terms of their time and space complexities.

Literature Review

Digraph matching algorithms such as the Similarity Flooding[1] algorithms, VF[5], VF2[2], Ullman, Schmidt and Druffel algorithm[5] and the Nauty[5] are used in Computer Science to evaluate graph isomorphism and sub-graph isomorphism problems on related graphs[1].

The solutions offered by most of graph matching algorithms are efficient, but because the algorithms approach the solution differently and are developed for different motives that solve the same problem, their degree of efficiency and quality of the solution is different.

The Similarity Flooding algorithm (SF) is a graph matching that was proposed by Sergey Menlik, Hector Garcia-Molina and Erhard Rahm [1], this is one of the interesting algorithms that is evaluated in this paper.

The SF algorithm works on two graphs (schemas, catalogues, or other data structures) that are related to each by their attributes, and it produces a multi-mapping of the corresponding nodes from the original input graphs [3]. The algorithm also uses filters that are applied to the resultant mapping that is produced by the algorithm, the purpose of the filters is to evaluate and produce the best mapping from the original mapping that was produced. The final, best mapping result from the algorithm is then reviewed, and if necessary, the results are adjusted [1].

What makes this algorithm effective with regards to graph matching is that, the algorithm itself is very versatile[1] and extensible[3] in that it only requires a general network representation of the of the graphs in order to perform the graph matching computation.

Another interesting graph matching algorithm that is analysed and studied in this paper is the VF2 algorithm [2]. The VF2 algorithm is an improvement of its predecessor, namely the VF algorithm[6]. The VF2 algorithm uses a more superior data structure for optimizing the matching time then the VF algorithm that is reported in [6].

The VF2 algorithm was introduced by L.P.Cordella, P.Foggiaa, C.Sansone and M.Vento. The algorithm is suitable for graph matching and isomorphic determination on large graphs, because its memory efficiency is significantly better then algorithms of the same kind [2]. In the VF2 algorithm, the mapping procedure is described using Space State Representation (SSR) that is described in [7]. Each mapping process has a state s, and the state is associated with a partial mapping solution M(s). The M(s) contains only the a subset of the components of the mapping function M. A partial M(s) identifies two sub graphs, one from Graphs A and another from B, namely GraphA(s) and GraphB(s). They are obtained by selecting the nodes that are in M(s) and their branches from Graph A and B.

The algorithm uses a feasibility function [2] that prunes the search tree constructed using the graphs. If the value returned by the function is true, it guarantees that state’ s', which is obtained from adding the partial mapping solution M(s) and mapping solution M to the state s is a partial isomorphism in s. Thus the final state is either an isomorphism between GraphA and GraphB, or sub graph isomorphism between GraphA and GraphB[2].

One of the most widely known graph matching algorithms is the Ullman's algorithm [8]. The Ullman algorithm detects isomorphism on graphs as well as in sub graphs of graphs [8]; the algorithm accomplishes this by employing a backtracking procedure with an effective look-ahead function to reduce the search space [8].

Though the algorithm may be old, it is still amongst the most exact graph matching algorithms[9].This is due to the fact that the algorithm's generality and its effectiveness with regards to graph matching[2], the algorithms is also among the fastest algorithms for the sub graph isomorphism problem.

The algorithm uses a model graph G, as a base of comparison. It takes in an input graph that is to be compared with the model graph denoted as G1. The adjacency matrix of the model graph is M, thus it is an nxn array. The adjacency matrix of the input graph is M1, thus it is an mxm array, and the permutation matrix of the two matrices in denoted as P. The algorithm's computation is based on the notion of finding all the sub graphs isomorphism’s by gradually setting the permutation matrix P, it does this by using backtracking recursively through the graphs and setting the associated elements indexes in P.

The algorithms that have been discussed perform syntactical (isomorphic) graph matching well and they do so very efficiently, but they do not perform semantic comparison well, if they do so at all. This paper aims to extend onto these algorithms to perform both the syntactical matches and the semantic matches were most popular graph matching algorithms are lacking.

Project Planning

The project time line outlines the projects progress throughout the year, from the 20th of March 2015 to the 15th of November 2015.

|  |  |
| --- | --- |
| **Date** | **Plan** |
| 24-Apr-15 | * Meeting with my supervisor to discuss the project proposal and progress for the past week and receive guidance on how to design and implement the Methodology. |
| 01-May-15 | * Meeting with my supervisor to discuss the designed methodology and the evaluation parameters that are used to compare the algorithms against each other. * Discuss the implemented algorithms and their results. * Start drafting the paper. |
| 08-May-15 | * Meeting with my supervisor to discuss the research that was done in the previous week and the algorithms that are mentioned in the paper as well as their relevance to the research. * Receive guidance on the paper and how to proceed with regards to writing. |
| 15-May-15 | * Meeting with my supervisor to discuss the results of the implemented algorithms as well as their graph matching quality as well as how they perform relative to each other with regards to graph matching quality. * Propose researched algorithms from the previous weeks and their relevance to the research. * Discuss the progress of the paper and a critical review from the supervisor. |
| 22-May-15 | * Meeting with the supervisor to discuss algorithms that have been implemented and their results relative to each other. * Discuss the quality of the evaluation of these algorithms, and discuss methods of improving the assessment of the algorithms. |
| 29-May-15 | * Meeting with my supervisor to discuss the research that was done in the previous week and the algorithms that are mentioned in the paper as well as their relevance to the research. * Propose researched algorithms from the previous weeks and their relevance to the research. |
| 04-Jun-15 | * Meeting with my supervisor to discuss the results of the implemented algorithms as well as their graph matching quality as well as how they perform relative to each other with regards to graph matching quality. * Discuss the progress of the paper and a critical review from the supervisor. |
| 11-Jun-15 | * Meeting with my supervisor to discuss the paper, and review the and time-line for the duration of the exam. * Discuss the progress of the paper and a critical review from the supervisor. |
| Preparations for Exams | |
| 17-Jul-15 | * Meeting with the supervisor to discuss algorithms that have been implemented and their results relative to each other. * Discuss the quality of the evaluation of these algorithms, and discuss methods of improving the assessment of the algorithms. |
| 24-Jul-15 | * Meeting with my supervisor to discuss the research that was done in the previous week and the algorithms that are mentioned in the paper as well as their relevance to the research. * Propose researched algorithms from the previous weeks and their relevance to the research. * Discuss the paper and receive a critical receive from my supervisor. |
| 31-Jul-15 | * Meeting with the supervisor to discuss algorithms that have been implemented and their results relative to each other. |
| 07-Aug-15 | * Meeting with my supervisor to discuss the progress of the paper and its quality. |
| 14-Aug-15 | * Present the paper and my finding to my peers and receive criticisms and advice on how to improve the quality of the paper. |
| 21-Aug-15 | * Meeting with my supervisor to discuss the quality of the paper and receive suggestions of improving the paper. * Discuss the evaluated algorithms and their results, and methods of improving the evaluation. |
| 28-Aug-15 | * Meeting with my supervisor to discuss the research that was done in the previous week and the algorithms that are mentioned in the paper as well as their relevance to the research. * Propose researched algorithms from the previous weeks and their relevance to the research. * Discuss the paper and receive a critical receive from my supervisor. |
| 04-Sep-15 | * Meeting with my supervisor to discuss the research that was done in the previous week and the algorithms that are mentioned in the paper as well as their relevance to the research. * Propose researched algorithms from the previous weeks and their relevance to the research. * Discuss the paper and receive a critical receive from my supervisor. |
| 11-Sep-15 | * Meeting with the supervisor to discuss the progress of the paper. |
| 18-Sep-15 | * Meeting with my supervisor to discuss the quality of the paper and receive suggestions of improving the paper. * Discuss the evaluated algorithms and their results, and methods of improving the evaluation. |
| 25-Sep-15 | * Meeting with my supervisor to discuss the research that was done in the previous week and the algorithms that are mentioned in the paper as well as their relevance to the research. * Propose researched algorithms from the previous weeks and their relevance to the research. * Discuss the paper and receive a critical receive from my supervisor. |
| 02-Oct-15 | * Present the paper and my finding to my peers and receive criticisms and advice on how to improve the quality of the paper. |
| 09-Oct-15 | * Meeting with my supervisor to discuss the quality of the paper and receive suggestions of improving the paper. * Discuss the evaluated algorithms and their results, and methods of improving the evaluation. |
| 16-Oct-15 | * Meeting with my supervisor to discuss the quality of the paper and receive suggestions of improving the paper. * Discuss the evaluated algorithms and their results, and methods of improving the evaluation. |
| 23-Oct-15 | * Meeting with my supervisor to discuss the quality of the paper and receive suggestions of improving the paper. * Discuss the evaluated algorithms and their results, and methods of improving the evaluation. |
| 30-Oct-15 | * Finalize the paper and results. * Discuss the paper with my supervisor. |
| 06-Nov-15 | * Meeting with my supervisor to discuss the quality of the paper and receive suggestions of improving the paper. * Present the paper and my finding to my peers and receive criticisms and advice on how to improve the quality of the paper. |
| 13-Nov-15 | * Final meeting with the supervisor and receive any final suggestions to improve the paper. |