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

Student number: 10637291
email: lectonlm@gmail.com
Supervisor: Dr Linda Marshall
Dept Computer Science, University of Pretoria

August 6, 2015

Abstract

In this paper, a variety of graph matching algorithms such as the Similarity Flooding algorithms and the VF2 are studied and compared against each in an attempt to measure their ability to perform both syntactic (structural) and the semantic (denotation) comparison efficiently. This evaluation analysis has practical implications in data warehousing, biochemical applications and e-business. The algorithms take two graphs as input, and they return an association between the two input graphs. As a result of the comprehensive analysis, we present an argument that considers the efficiency of the studied algorithms as well as the quality of their comparisons to determine which amongst them is the best to perform both syntactic and semantic comparison, and why the algorithm performs better than the others of its kind.

1 Introduction

Graphs are used in a variety of disciplines and they serve a multitude of purposes, such as representing the syntactical representations of the components in a sentence and their relationships to each other[10], describing the structures of chemical compounds in chemistry[11],representing the entries in a database and their relationships, pattern recognition and computer vision[2]. Thus because the digraph database structure is so widely used, it is important to find methods of finding the correlation between of data schemas in applications that make use of this comprehensive data structure[1]. This process is referred to as graph matching. Before we get to some of the graph matching algorithms, some concepts of graph theory must first be understood.

A Graph in Mathematics and Computer Science is a pair G = (V, E), where V is the set of vertices and E is the set of edges, formed by pairs of vertices. Figure 1 demonstrates the structural attributes of a graph.

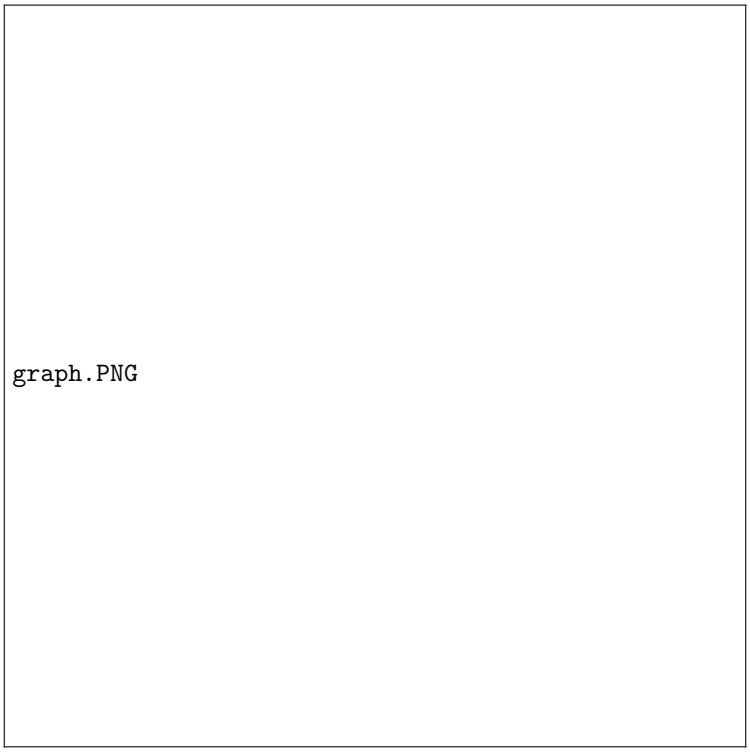

Figure 1: Representation of a graph

Figure 2: Representation of a digraph

The graphs discussed in this paper are called directed graphs, refer to figure 2. Digraphs are graphs whose edges have a certain direction in which they are going, in figure 2 this characteristic is demonstrated by the edge 1, that goes from node a to node b, and also by edges 3 that goes from node c to node a. Note that in figure 1, the graph is referred to as a bigraph because the direction of its edges are in both ways as it is not specified.

Two graphs are said to be isomorphic if they are syntactically similar to each other iff there is a bijection between their respective nodes which make each edge of G1 correspond to exactly one edge of G2, and vice versa[12], i.e. the graphs are structurally the same to each other. This property is demonstrated in figure 3. The two graphs look very different, but when they are further inspected, it is evident that the two are a representation of the same data scheme or even the same graph that has been rearranged.

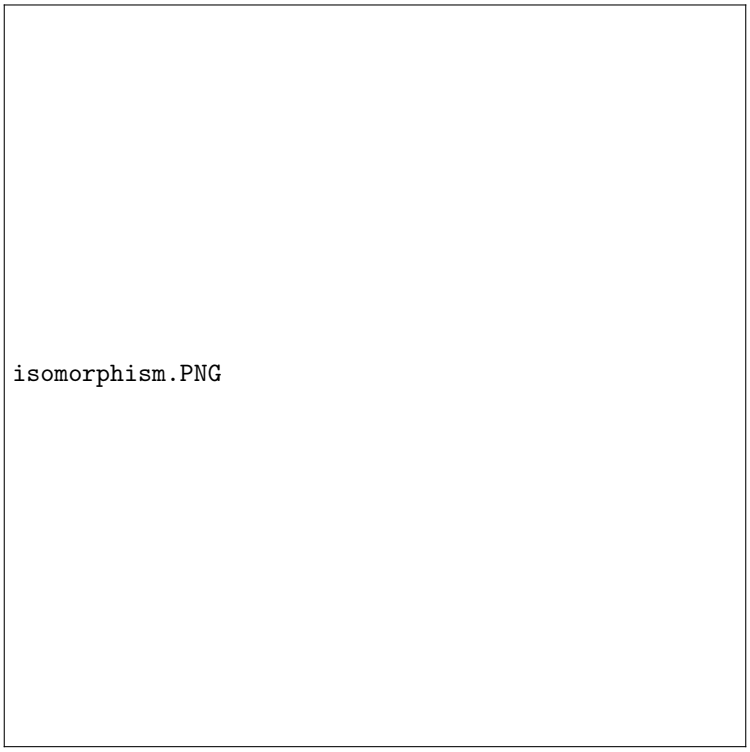

Figure 3: Representation of the isomorphism property

Graphs are often represented as a matrix, more precisely a adjacency matrix. This is a nxn matrix A, with A(i,j) = 1 iff(i,j) E[12]. This means that wherever there is an edge in the graph, it is denoted by a 1 in the matrix, places in the matrix A where there is an absence of an edge E are denoted by 0.

Figure 4 depicts the association between a graph and its adjacency matrix.

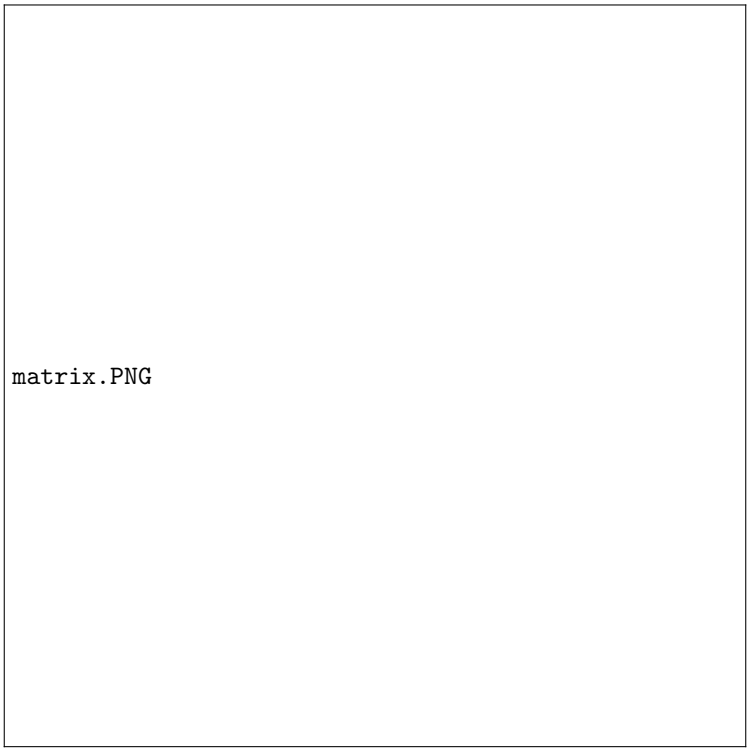

Figure 4: Representation of a graph and its associated adjacency matrix

Lastly we introduce the concepts of sub graphs in Graph Theory. A graph G' = (V',E') is a sub graph of a graph G = (V, E) if V' V and E' E[12], i.e. the graph G' must appear inside graph G in order to be considered a sub graph of G.

A graph matching algorithm computes the association between two graphs, namely graph G' and G" and measures the degree of this association.

This paper analyses some of the more popular graph matching algorithms such as the Similarity Flooding[1] algorithms, VF[5], VF2[2], Ullman, Schmidt and Druffel algorithm[5] and the Nauty[5], specifically those that are tailored (designed) for digraphs, though most of these algorithm only perform syntactical (isomorphism) comparisons, we are going to investigate those algorithms that also perform semantic (content in the graphs) comparison or can be modified so that the existing algorithms can also be extended to perform semantic comparison.

2 Problem Statement

Evaluating the isomorphic (syntactic) and the semantic relationship between two or more di-graphs using search algorithms, and then comparing the similarities of the graphs is an expensive enter-

prise. This is because the procedures that are required to achieve this task are often expensive and the large data sets that are used for semantic 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.

Graph matching algorithms such as the Similarity flooding algorithm[1] and the VF2 algorithm[2] compare the syntactic and semantic 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.

2.1 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 semantically 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.

2.2 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.

3 Methodology

Primary Research Method: Experiments **Secondary Research Method:** Arguments

3.1 Experiment

3.1.1 Data set

The data set that is used in this experiment is a set of digraph objects of an arbitrarily large size. The digraphs that are used meet the following conditions

- They are either partially or completely isomorphic in relation to one or other digraph inside the set.
- They are either partially,completely or not semantically similar to one or other digraph inside the set.

A quantitative approach is employed in evaluating the algorithms against each other. The search graph matching algorithms are implemented and experimented on, using two digraph objects at a time. The experiment for that is employed measure and records the algorithms ability and performance of the algorithms in cases where Graph A and Graph B are either partially or completely isomorphic, and cases where Graph A and B are partially, completely or not semantically similar.

A measure of the degree of similarity for both the syntactical and semantically similarity is calculated and recorded for each test case.

3.1.2 Experiment implementation

We will refer to Graph A and Graph B in this context.

The algorithm has to insert an arbitrarily large number of nodes.

(1) The two graph are compared to each other to evaluate the degree of their similarity.

(2) The comparison that is done is dependent on the relationship between Graph A and B.

(I) **Syntactic comparison**

(A) Graph A is compared with the whole of Graph B for syntactical similarity.

(i) A new subgraph of Graph B is generated.

(ii) The subgraph is compared with graph A for syntactical similarity.

(a) The time taken to perform the comparison is recorded.

(b) The amount of memory used by the algorithm to perform the comparison is recorded.

(c) The degree of syntactical(isomorphic) similarity is calculated relative to the graph of B and stored as well as the subgraph. .

(iii) Once all the subgraphs of Graph B have been experimented upon, the result of the subgraph with the best degree of syntactical(isomorphic) similarity is recorded together with the subgraph.

(B) If Graph A and B are not completely syntactically similar, Graph A is compared with all the possible subgraphs of Graph B.

(II) **Sementic comparison**

(A) The sementic comparison is performed on Graphs that are Syntactically similar to some degree, this is because there is no point in comparing the sementics relationship of a Graphs if they are not even structurally the same.

(i) If the syntactical similarity of Graph A and Graph B is equal to 1.0, then...

(a) Graph A is compared with the whole of Graph B for semantical similarity.

(1) The time taken to perform the comparison is recorded.

(2) The amount of memory used by the algorithm to perform the comparison is recorded.

(3) The degree of semantical(isomorphic) similarity is recorded.

(ii) If the syntactical similarity of Graph A and Graph B is a fraction of 1.0, then...

(a) Graph A is compared with all the possible subgraphs of Graph B.

(1) The subgraph that was recorded during the Syntactic comparison is retrieved.

(2) The subgraph is compared with graph A for semantical similarity.

(I) The time taken to perform the comparison is recorded.

(II) The amount of memory used by the algorithm to perform the comparison is recorded.

(III) The degree of semantical(isomorphic) similarity is calculated relative to the graph of B and recorded as well as the subgraph.

Calculating the Syntactic and Sementical similarities

The Syntactic and Sementical similarities are a calculated relative to some graph, and they both use the same function.
The funation calculates all the nodes in the reference graph, and they are divided by the number of nodes inside the subgraph.
Example.

(a) Reference Graph (b) Sub graph

Figure 5: A figure with two graphs that are used to demonstrate the calculation

Similarity = 3/7 = 0.429 units.

3.2 Experiment

The recorded results from all the algorithms is analyzed and used to identify which amongst the algorithms is the best and for which cases it is the best in.
The results from the algorithms are weighed against a comparison criteria, and it is this criteria that is used to evaluate the algorithms against each other.

3.2.1 The comparison criteria

The criteria that the algorithm results are weighed against is specified below.

(1) Space efficiency

(2) Complexity of the algorithm

(3) Degree of synactical similarity result generated by the algorithm per test case.

(4) Degree of sementical similarity result generated by the algorithm per test case.

Once the results have been obtained and throughly evaluated, then the algorithms with the best perform per criteria field are recorded.
And from that set, the best algorithm is chosen overall relative to the others based on the criteria.

4 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 isomorphisms 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.

5 Planning

The use case are depicted in the table below.

Task	**Start Date**	**End Date**	**Date Completed**	**Days remaining**

• 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. • 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.	24-Apr-15	15-May-15		21
Receive guidance on the paper and how to proceed with regards to writing.	15-May-15	22-May-15		28

• 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. • 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.	22-May-15	04-Jun-15		41

• 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. • 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. • Meeting with my supervisor to discuss the paper, and review the and timeline for the duration of the exam. • Discuss the progress of the paper and a critical review from the supervisor.	04-Jun-15	11-Jun-15		48
Preparations for Exams				

• 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. • 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.	17-Jul-15	31-Jul-15		98

• Meeting with the supervisor to discuss algorithms that have been implemented and their results relative to each other. • Meeting with my supervisor to discus the progress of the paper and its quality. • Present the paper and my finding to my peers and receive criticisms and advice on how to improve the quality of the paper. • 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.	31-Jul-15	21-Aug-15		119

• 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. • 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.	28-Aug-15	04-Sep-15		133

• 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. • Meeting with my supervisor to discuss the research that was done in the previous week and he 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. • Present the paper and my finding to my peers and receive criticisms and advice on how to improve the quality of the paper. • 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.		18-Sep-15	09-Oct-15		168

• 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. • 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	23-Oct-15		182
• Finalize the paper and results. • Discuss the paper with my supervisor.	30-Oct-15	01-Nov-15		191
• 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.	01-Nov-15	03-Nov-15		193

Final meeting with the supervisor and receive any final suggestions to improve the paper.	03-Nov-15			

References

[1] S. Melnik, H. Garcia-Molina and E. Rahm, *Similarity flooding: a versatile graph matching algorithm and its application to schema matching*, in Proceedings 18th ICDE, San Jose CA, Feb 2002.

[2] L. Cordella, P. Foggia, C. Sansone, and M. Vento, *A (sub)graph isomorphism algorithm for matching large graphs*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 10, pp. 13671372, Oct. 2004.

[3] Marshall, B., Chen, H., Madhusudan, T. (2006), *Matching knowledge elements in concept maps using a similarity flooding algorithm. Decision Support Systems*, 42 (3), 1290-1306.

[4] *VF2 Algorithm(2015)*, Available at: `https://networkx.github.io/documentation/latest/reference/algorithms.isomorphism.vf2.html`, (Accessed: 10 April 2015).

[5] P. Foggia, C. Sansone, M. Vento, *A performance comparison of five algorithms for graph isomorphism*, submitted for publication to the Third TC-15 International Workshop on Graph Based Representation, Italy, 2001.

[6] L.P. Cordella, P. Foggia, C. Sansone, M. Vento, *Evaluating Performance of the VF Graph Matching Algorithm, Proc. of the 10th International Conference on Image Analysis and Processing*, IEEE Computer Society Press, pp. 1172-1177, 1999.

[7] N.J. Nilsson, *Principles of Artificial Intelligence*, Springer-Verlag, 1982.

[8] J.R. Ullmann, *An Algorithm for Subgraph Isomorphism, Journal of the Association for Computing Machinery*, vol. 23, pp. 31-42, 1976.

[9] B. T. Messmer, *Efficient Graph Matching Algorithms for Preprocessed Model Graphs*, Ph.D. Thesis, Inst. of Comp. Sci. and Applied Mathematics, University of Bern, 1996.

[10] H. Ehrig, *Introduction to Graph Grammars with Applications to Semantic Networks*, Comput. Math. Appl. 23, pp. 557-572, 1992.

[11] D.H. Rouvray, A.T. Balaban, *Chemichal Applications of Graph Theory in Applications of Graph Theory*, Academic Press, New York, pp.177-221, 1979.

[12] Reinhard Diestel, *Graph Theory*, Graduate Texts in Mathematics, Vol. 173, Springer Verlag, Berlin, 1991.