INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

Achieving Polynomial Time Complexity in Maximum Common Subgraph (MCS) Computation with reference to Text Modeling

Kaushik Kishore Phukon*

1 Department of Information Technology, Pandit Deendayal Upadhyaya Adarsha Mahavidyalaya, Tulungia, Bongaigaon, Assam-783383, India

OPEN ACCESS

Received: 27/05/2025

Accepted: 08/07/2025

Published: 29/07/2025

Citation: Phukon KK (2025) Achieving Polynomial Time Complexity in Maximum Common Subgraph (MCS) Computation with reference to Text Modeling. Indian Journal of Science and Technology 18(29): 2347-2352. https://doi.org/10.17485/IJST/v18i29.973

* **Corresponding author**. kaushikphukon@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: To make representation of text through directed labeled graph and also to achieve Maximum Common Subgraph (*MCS*) computation in limited time for the purpose of measuring similarity among the graphical objects. **Method:** This work has employed the maximum common subgraph approach for measuring the similarity in between two objects under representation. **Findings:** The proposed approach is successful in achieving polynomial time complexity, which is theoretically established in this article. The new similarity measure so developed is also more effective and efficient than the prevailing method and is shown with the help of an arbitrary example. **Novelty:** Achieving polynomial time complexity for the maximum common subgraph problem is a significant challenge in graph theory and algorithm design. This research has demonstrated how polynomial time complexity can be achieved in case of maximum common subgraph computation problem with directed and completely labeled graph in a methodical way.

Keywords: Time Complexity; Text Representation; Graph; MCS; Word Net

1 Introduction

Text modeling is a process in natural language processing (NLP) that involves developing algorithms or models capable of understanding, generating, and working with text. It is a fundamental part of many AI-driven applications like chatbots, machine translation, sentiment analysis, and text generation. At its core, text modeling deals with representing textual data in ways that machines can interpret, analyze, and process. Natural language texts have information meant for general-purpose person to person conversation. Hence, its structure follows the syntactic rules used to create these conversations. We generally do not consider natural language text as character sequences. We consider them as a sequence of groups of alphabetic symbols forming words. Hence the character-based representation techniques are barely able to capture and use long-range correlations existing between words. However, if text representation techniques use bigger units than single characters as the basic storage element, they

would be able to capture and use long-range correlations [1]. Thus, a word-based algorithm is a batter choice when a natural text is represented, as words are the true reflector of text entropy [2]. In this context graphs are the right scientific structure that can represent natural language text in a meaningful way.

The first and foremost activity in the text modeling process is the representation of the text with the help of an appropriate model that can mirror the text or the page containing the text. The models reflect the data objects and take care of the intended purpose. After the representation step, we need to measure the similarity or dissimilarity between the data objects in the representation space. The text modeling or mining processes are the procurer that characterizes a specific clustering task to achieve certain predefined objectives.

1.1 The Spectrum of text modeling

In a text modeling method, all features are generally not selected. The features are selected based on some established criteria such as inverse document term frequency, simple document term frequency, mutual information, information gain, term strength, etc. [3]. A preprocessing step is also applied before the feature selection process with an intention to remove the non-significant words that are believed to have no impact in the mining process. For example, the stop words such as the, and, an, etc. are presumed to have no significance in capturing meaningful information. Although the word frequency based approaches are very popular and widely used, the approaches are not recognized as perfect way to model text. As stated, this text modeling method considers text as a collection of some dictionary words and do not brother about the contextual organization and the ideas conveyed by the text. As a result, the similarity measures based on such a representation model cannot perceive contextual similarity due to the lack of contextual information [4]. Significant attempts have been made to discover some new way to represent text apart from the term frequency approach. Some of the significant efforts are N/grams, Bigrams, the extraction of words semantic relations through corpus statistics, the use of background knowledge by replacing words with their higher concepts in an ontology, and the consideration of word sequences with the help of a graphical model [5]. We found six different graph models for representing web documents in the literature. They are: standard, simple, n-distance, n-simple distance, absolute frequency, and relative frequency. Three other models based on these fundamental models are also found namely TSGM (Tag Sensitive Graph Model), CSGM (Context Sensitive Graph Model) and an enhanced version of CSGM [6]. In this research article we are considering the enhanced version i.e. the $CSGM_2$ text modeling approach to represent textual data.

Numerous research works have been perused in the area of graph similarity in order to incorporate the additional information allowed by graph representations. The first step in this regard is the formulation of mathematical frameworks for dealing with graphs [7]. In the literature, the work comes under several different topic names including graph distance, graph matching, inexact graph matching, error-tolerant graph matching or error-correcting graph matching. In case of exact graph matching, two or more graphs are matched to determine if the graphs are identical or not. In case of inexact graph matching, the aim is not to find a perfect match but to find a ”best” or ”closest” match. Graph distance is a numeric measure of dissimilarity between graphs, with larger distances implying more dissimilarity [8]. By graph similarity, we mean we are interested in some measurement that tells us how similar graphs is regardless of if there is an exact matching between them.

2 Methodology

2.1 Word Net Building

The $CSGM_2$ technique transforms the original text into a word net (graph) in which all relationships between adjoining words are retained. For all kinds of natural language, all words in a sentence have some relationship to all other words in it. The closer two words are to each other; the stronger their connection tends to be [9] and graphs are the right scientific structures to represent such relationship.

Suppose $CSGM_2$ technique performs on a text T of length t words, drawn from a vocabulary $\sum$ containing σ different words: To implement the word net a digraph G: = (V, E) is considered in accordance with the following features:

1. Each unique term appearing in the document becomes a node in the graph representing that document. Each node is labeled with the term it represents and a global identifier, which is a non-negative integer value $x \in [1, \sigma]$ where σ is the total word count in T. The node labels in a document graph are unique, since a single node is created for each keyword even if a term appears more than once in the text.
2. Second, if word *a* immediately precedes word *b* somewhere in the document, then there is a directed edge from the node corresponding to term *a* to the node corresponding to term *b* with an edge labeled as (d:i), where d is a user specified parameter representing the distance between the two words and i is the number of times that the edge has been traversed.

3. There are no multiple edges in G. If there is more than one transition between two consecutive words ω_x and ω_y (represented in x and y vertices respectively) only a single edge, (x, y), is modeled.

Thus, a word net, implemented through the diagraph G: = (V, E), is adaptively built as T is processed. Graph and text representations are coordinated at the same time as the word net is incrementally built and updated. The processing of the text T at any specific point of time can be illustrated as given below.

We keep c ∈ V as the current vertex (c stores the word say $\omega_c \in \sum$) and suppose that the word ω is reached. The three following situations can happen:

1. If ω is reached for the first time a vertex v is appended to V for storing this new word, and a new edge (c, v) from the current vertex is inserted in E. This edge represents the word transition from ω_c to ω
2. If ω is previously appeared (it is stored in v ∈ V), but it was never preceded by ω_c. Then a new edge (c, v) must be appended to E to represent this new word transition.
3. If ω previously appeared and at some time, was preceded by the word represented in c then it implies that the edge (c, v) ∈ E, and it is locally identified in c with a non-negative value x. The statistical information of (c, v) is only updated.

An algorithm formalizing the transformation process can be seen in ref.[10]. With the help of the algorithm, we can transform a text document into a graph representation by retaining all necessary information required for further processing. Now to calculate the distance between two or more documents in graphical format we are considering the maximum common subgraph approach[11]. The calculation of distance or similarity between the documents under consideration is the vital step towards categorization or clustering[12].

2.2 Maximum Common Subgraph Approach

Bunke has shown[13] that there is a direct relationship between graph edit distance and the maximum common subgraph between two graphs. Specifically, the two are equivalent under certain restrictions on the cost functions. A graph *g* is a maximum common subgraph of graphs G_1 and G_2, denoted *MCS(G_1,G_2),* if:

1. g ⊆G_1
2. *g*⊆G_2 *and*
3. there is no other subgraph $g'(g' \subseteq G_1, g' \subseteq G_2)$ such that $|g'|>|g|$

Here $|g|$is usually taken to mean |V|, *i.e.* the number of nodes in the graph. It is used to indicate the ”size” of a graph. Methods for determining the *MCS* are given in refs.[14],[15]. The general approach is to create a compatibility graph for the two given graphs, and then find the largest clique within it. He also showed that, when computing the editing matching function based on graph edit distance, the function with the lowest cost is equivalent to the maximum common subgraph between the two graphs under certain conditions on the cost coefficients. This is intuitively appealing, since the maximum common subgraph is the part of both graphs that is unchanged by deleting or inserting nodes and edges. To edit graph G_1 into graph G_2, one only needs to perform the following steps:

1. Delete nodes and edges from G_1 that don't appear in *MCS(G_1,G_2)*
2. Perform any node or edge substitutions if necessary
3. Add the nodes and edges from G_2 that don't appear in *MCS(G_1, G_2)*

Following this observation that the size of the maximum common subgraph is related to the similarity between two graphs, Bunke and Shearer have introduced a distance measure based on *MCS* as stated below:

$$dist_{MCS}(G_1, G_2) = 1 - \frac{|MCS(G_1, G_2)|}{\max(|G_1|, |G_2|)} \tag{2.1}$$

where max(x, *y)* is the usual maximum of two numbers *x* and *y,* and |...| indicates the size of a graph (usually taken to be the number of nodes in a graph).

3 Results and Discussion

3.1 The New Distance Measure Based on CSGM$_2$

The maximum common subgraph approach is an effective and efficient way to calculate graph distances. But with CSGM$_2$ model, the existing method for determining *MCS* will not work well as the method is based only on number of nodes of the graphs under consideration. As the CSGM$_2$ model is capable to hold more information than that of node information only, so to have full benefit from the CSGM$_2$ model the following *MCS* distance measure has been developed.

$$dist_{MCS}(G_1,G_2) = 1 - \left(\frac{\sum_{SOM} d^{\pm}(MCS(G_1,G_2))}{\max(\sum d^{\pm}(G_1),(\sum d^{\pm}(G_2)))} \right) \tag{3.1}$$

where $\sum d^{\pm}$ represents the sum of in-degree and out-degree of the directed graph and ($\sum_{SOM} d^{\pm}$) represents the sum of the minimum of the degrees generated by each common node of the graphs which are included in the *MCS*. In case when there are two or more *MCS* then we should consider max ($\sum_{SOM} d^{\pm}$), where max(x, *y)* is the usual maximum of two numbers *x* and *y*.

The embodiment of in degree and out degree and edge labels into the computation process significantly improves the efficiency and effectiveness of the distance measure. The efficiency and effectiveness achieved through this incorporation is explained below with the help of an arbitrary and simple example.

Figure 1 reveals the greater accuracy in calculating graph distance in between two arbitrary graphs in comparison to existing method [5] of calculating graph distance. This is achieved through utilization of necessary information by considering in and out degrees of the respective directed graphs.

3.2 Steps for Extracting the MCS

Calculating the distance between two graphs (**Eq. 3.2**) requires the computation of the maximum common subgraph of the pair of graphs. The determination of the maximum common subgraph in the general case is known to be an NP-complete problem. However, with our graph representation for textual documents, each node in a graph has a unique label (representing a unique term) that no other node in the graph has. Thus, the maximum common subgraph, G_{MCS}, of a pair of graphs, G_1 and G_2, can be computed by the following procedure:

1. Find the nodes V_{MCS} by determining the subset of node labels that the original graphs have in common with each other and create a node for each common label.
2. Find the edges E_{MCS} by examining all pairs of nodes from step (1) and introduce edges that connect pairs of nodes in both of the original graphs.
3. Select the *MCS* which have the highest no of nodes and

$$\sum_{SOM} d^{\pm} = \max\left(\sum_{SOM} d^{\pm} MCS(G_1,G_2) \right).$$

We can observe the computational complexity of different steps of the above stated procedure as follows:

For step 1:

Here we need only to compare each node label from one graph to each node label of the other and determine whether there is a match or not. Let us suppose that graph G_1 has V_1 number of nodes and graph G_2 has V_2 number of nodes. Thus, the maximum number of comparison will be $|V_1| \cdot |V_2|$, and since each node has a unique label we only need to consider each combination once. Thus, the overall complexity will be-

$$O(f_1(n)) = O(|V_1| \cdot |V_2|) \tag{3.2}$$

For step 2:

From step 1 we will have V_{MCS} number of nodes i.e. the number of common nodes that both the graphs have. Now we must look at all combinations of pairs of nodes to determine if an edge should be added between them or not. This computational complexity can be mathematically expressed as.

$$\left(|V_{MCS}|_{C_2}\right) = \frac{|V_{MCS}|!}{2!\,(|V_{MCS}|-2)!} = \frac{|V_{MCS}|.(|V_{MCS}|-1)}{2} < |V_{MCS}|^2$$

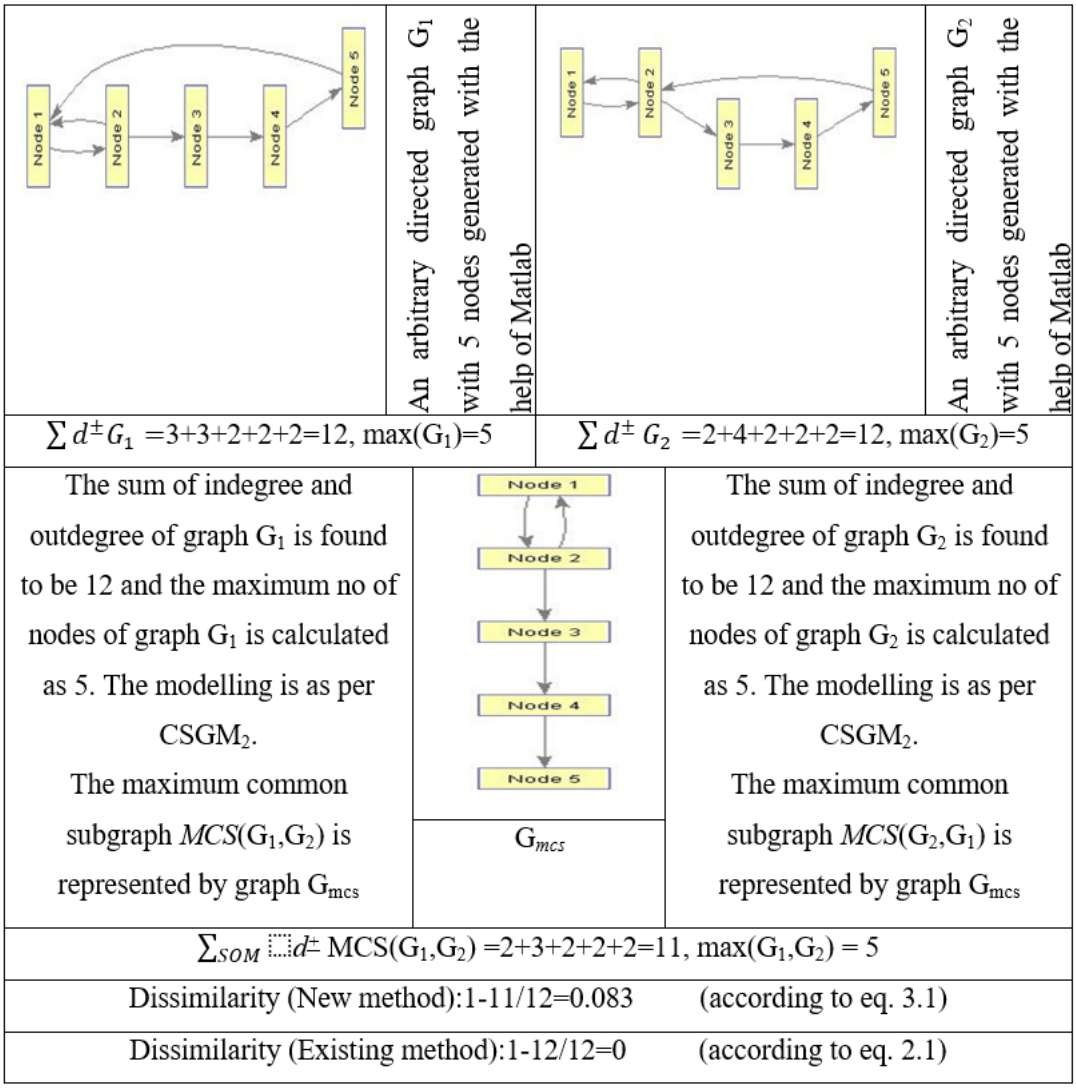

Fig 1. Calculation of dissimilarity among graphical objects

Thus, we get the computational complexity of step 2 as

$$O(f_2(n)) = O(|V_{MCS}|^2) \tag{3.3}$$

So, the overall complexity becomes:

$$O(f_1(n)) + O(f_2(n)) = O(|V_1| \cdot |V_2| + |V_{MCS}|^2) \leq O(|V|^2 + |V_{MCS}|^2) = O(|V|^2) \tag{3.4}$$

if we substitute V = max($|V_1|$), ($|V_2|$)

From the **eq. 3.4** it can be observed that the computational complexity for calculating the *MCS* is reduced to polynomial time with directed and completely labeled graphs or more specifically while using the $CSGM_2$ representation to represent text.

4 Conclusion

Achieving polynomial time complexity in Maximum Common Subgraph (*MCS*) computation represents a significant advancement, particularly in domains where structural similarity plays a critical role-such as text modeling. Traditional *MCS*

algorithms often face scalability challenges due to their NP-hard nature, making them computationally expensive for large datasets. This research work provides a way to represent textual documents through directed, completely labeled graphs that can retain all required structural and semantic information that is necessary to recreate the original document with desired level of accuracy and way to measure similarity among them in polynomial time. However, by leveraging domain-specific constraints, approximate methods, or heuristics derived from the characteristics of textual data (such as syntactic or semantic similarity graphs), it becomes possible to reduce the search space and achieve near-polynomial or practical polynomial performance. In the context of text modeling, this facilitates more efficient comparisons of document structures, topic graphs, or semantic networks, thereby enabling scalable applications in areas such as plagiarism detection, document clustering, and information retrieval. While exact polynomial-time solutions for general *MCS* remain elusive, focused approaches aligned with text modeling tasks provide promising pathways toward feasible and impactful solutions.

References

1) Horspool R, Cormack G. Constructing Word-Based Text Compression Algorithms. 1992. https://doi.org/10.1109/DCC.1992.227475.
2) Salomon D. Data Compression: The complete reference. 2004. Available from: https://books.google.co.in/books/about/Data_Compression.html?id=FlWjiShUst0C&redir_esc=y.
3) Mirończuk M, Protasiewicz J. A recent overview of the state-of-the-art elements of text classification. https://doi.org/10.1016/j.eswa.2018.03.058.
4) Fuchs M, Riesen K. Fast approximate maximum common subgraph computation. *Pattern Recognition Letters*. 2025;190:66–72. https://doi.org/10.1016/j.patrec.2025.02.006.
5) Schenker A, Bunke H, Last M, Kandel A. Graph Theoretic Techniques for Web Content Mining;vol. 62. World Scientific Publishing Co. Pte. Ltd.. 2005. Available from: https://www.scribd.com/document/859592694/19036.
6) Phukon KK, Baruah HK. A Graph Theoretical Preprocessing Step for Text Compression. *International Journal of Multimedia and Ubiquitous Engineering*. 2015;10(5):263–276. https://dx.doi.org/10.14257/ijmue.2015.10.5.24.
7) Hammouda K, Kamel M. Phrase-based document similarity based on an index graph model. 2002. https://doi.org/10.1109/ICDM.2002.1183904.
8) Cardone L, Quer S. The Multi-Maximum and Quasi-Maximum Common Subgraph Problem. *Computation*. 2023;11(4):69. https://doi.org/10.3390/computation11040069.
9) Martinez-Prieto M, Adiego J, Fuente P. Natural Language Compression on Edge-Guided Text Preprocessing. . Available from: https://dl.acm.org/doi/10.5555/1778666.1778668.
10) Phukon KK. Incorporation of contextual information through Graph Modeling in Web content mining. *Indian Journal of Science and Technology*;13(46):4573–4578. https://doi.org/10.17485/IJST/v13i46.1660.
11) Yu K, et al. Fast Maximum Common Subgraph Search: A Redundancy-Reduced Backtracking Approach. 2025. Available from: https://arxiv.org/abs/2502.11557v1.
12) Quer S, Marcelli A, Squillero G. The Maximum Common Subgraph Problem: A Parallel and Multi-Engine Approach. *Computation*. 2020;8(2):48. https://doi.org/10.3390/computation8020048.
13) Bunke H. On a Relation between Graph Edit Distance and Maximum Common Subgraph. *Pattern Recognition Letters*;18:689–694. https://doi.org/10.1016/S0167-8655(97)00060-3.
14) Levi G. A Note on the Derivation of Maximal Common Subgraphs of two Directed or Undirected Graphs. *Calcolo*;9:341–354. https://doi.org/10.1007/BF02575586.
15) McGregor JJ. Backtrack Search Algorithms and the Maximal Common Subgraph Problem. *Software—Practice and Experience*;12:23–34. https://doi.org/10.1002/spe.4380120103.