# An Effective Approach for Solving Subgraph Isomorphism Problem

Zong Ling

Department of Electrical Engineering, University of Hawaii (UH)
493 Holmes Hall, 2540 Dole Street, Honolulu, Hawaii 96822, U.S.A.
Phone: (808) 956-7249; Fax: (808) 941-1399; Email: ling@spectra.eng.hawaii.edu

**Abstract:** A novel and effective approach is presented in this paper for solving the well-known difficult problem of subgraph isomorphism. By integrating planning strategies with the concept of centered spanning tree, the presented approach achieves an average-case, near-linear performance. A majority of practical subgraph isomorphism problems deals with the cases of labeled graphs, which contains the NP-complete case of unlabeled graphs. This meaningful relationship between the sizes of label sets and graphs is exploited by both analyses and experimental results for use in the prediction of performance.

***Keywords***: Subgraph Isomorphism, Planning, Algorithms, Pattern Recognition, Search

## 1. INTRODUCTION

The SubGraph Isomorphism (SGI) problem is a frequent computation in many applications, where the search and recognition of a smaller graph from a larger graph is needed. For instances, 1) In bioinformatics [1], it is always required to search an isomorphic special piece of protein sequence (3D labeled graph) in the Protein Data Bank (PDB) with the query protein sequence for further biological analysis; 2) In robot visions [11], it needs to recognize 3D objects (based on the line-drawings) by isomorphically matching with their canonical models in a computer model base; 3) In VLSI computer-aided design (CAD) [7], one of the critical problems is that of determining whether layout of the circuit geometry corresponds to the specification of circuit [4], especially to find the subcircuits from a lager circuit [8].

Previous approaches to solve SGI problem are predominantly based on tree search approach such as J. R. Ullmann's depth-first algorithm [10], which is still in use today in the popular software package *POSSUM* (Protein On-line Substructure Searching-Ullmann Method) for protein sequence comparison, as well as Corneil's breadth-first algorithm [2], which is presently the core component of *Gemini* and *SubGemini* -- the best performing package today for subcircuits extraction in VLSI layout verification [8]. However, these algorithms quickly become ineffective when graph size becomes larger, as predicted by the NP-completeness of SGI. On the other hand, the needs from application areas, old and new, are pushing for increasingly larger graph sizes. In applications, such as subcircuit verification and molecular structure analysis, typical problem sizes have already grown to the order of thousands of nodes and edges.

Since SGI problem is a classical constraint satisfaction problem, in this paper, we propose the alternative of transforming it into the resource management paradigm. Based on a set of strategies and centered spanning tree for graph decomposition and partition, the presented approach, following the principles of the *Constrained Resource Planning* (*CRP*) model [6], is shown to achieve near-linear performance on the average.

## 2. PROBLEM DESCRIPTION

SubGraph Isomorphism is a well known concept -- a given graph $G_1$ is isomorphic to a subgraph H of another given $G_2$ if there exists an one-to-one mapping of the nodes of $G_1$ onto the nodes of H such that all corresponding edge adjacencies are preserved. The problem is that of finding an *efficient* algorithm or approach for determining whether one given graph (*pattern graph*, $G_1$) is an isomorphic subgraph of another graph (*model graph*, $G_2$). Here, all graphs or subgraphs are supposed to be connected graphs.

SGI problem is not only of considerable practical importance but also of theoretical interest due to its known NP-complete complexity. Its difficulty can be seen easily from the fact that selecting out of the $m^n$ possibilities that arise in the combinatorial matching of $n$ nodes in the smaller graph to $m$ nodes in the larger graph, while preserving all the adjacencies [9]. No efficient algorithm for this problem is known so far [3], and it was conjectured by many experts that no polynomial-time algorithm exists because of its NP-completeness [5][9].

Generally, for majority of practical applications, both pattern and model graphs are labeled graphs. Their label sets are known and their sizes also. In the extreme case of the label set size being one, a labeled graph becomes indistinguishable with an unlabeled graph, so that labeled SGI problem contains unlabeled SGI problem which is a well known NP-Complete problem. In fact, unlabeled SGI problems are usually used on theoretical analysis since it can provide an up-bound of time complexity due to its NP-completeness. Therefore, in this paper, the labeled graph we concerned below would not lose the generality of SGI problem and should be more appropriate to meet the practical requirements.

## 3. OUR APPROACH

As shown in Figure 1, our approach for solving SGI problem contains two parts: preprocessing and resource management.

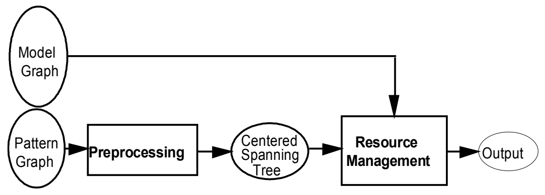

**Figure 1. Our Approach**

In pre-processing, we will extract a centered spanning tree from pattern graph and then partition model graph into small bounded areas. The matching procedure which is treated as a constraint resource planning procedure, thereby, can be limited within the bounded areas

### 3.1 Preprocessing

There are two main operations in preprocessing, 1) extract a centered spanning tree from pattern graph; 2) partition model graph.

3.1.1 Tree Extraction

Assume that G is a labeled and connected graph.

3.1.1.1 *Terminology*

Following graphic concepts will be utilized.

*Distance $d(n_1, n_2)$*: of two nodes $n_1$ and $n_2$ is the length of shortest path between them.

*Eccentricity E(n)*: of a node $n$ is the distance from $n$ to the nodes farthest from $n$ in G; that is, $E(n) = \max d(n, n_i), n_i \in G$

*Center*: the node with minimum eccentricity in G.

*Radius*: the eccentricity of a center.

3.1.1.2 *Construction of Centered Spanning Tree*

To locate a center of graph G, all shortest paths in G are calculated. And then the maximal length value of the shortest paths (or summary of the shortest paths) from each node to other nodes can form a list. The minimal element in the list corresponds to the center node of G. Starting from the center nodes (as root), a centered spanning tree of $G_1$ is constructed by breadth-first search algorithm. Then, $G_1$ is decomposed into two parts: a Centered Spanning Tree (CST) and a CoTree (complement part of the CST in G). Actually, CST can be constructed by either depth-first or breadth-first search algorithm. Since we need to utilize the height of CST as the radius on partitioning model graph and expect to obtain the minimal radius which represents the range of bounded area, breadth-first search which can minimize the height of a tree is our preference.

3.1.2 Partition of Model Graph

First of all, in $G_2$, each node with the same label as and larger degree than that of the center node (root of CST) in $G_1$ is chosen as the starting point. Then, around the starting points, the small areas are determined respectively according to the height (radius) of CST. Within each of the bounded areas, which is much less than the original $G_2$ in average, the resource management approach is then used for isomorphic matching.

### 3.2 Resource Management

From the perspective of operational research, the matching procedure in SGI can be treated as an instance of resource management. It is a process of task and solution planning directly associated with the concern of resource utilization and conservation.

3.2.1 Edge Units

An edge unit (EU) is defined as an edge associated with its two terminal nodes in a graph. As the fundamental matching elements, EUs can not only preserve node connections which are the requirement of isomorphism but also extend the label attribute scope from |NLS|+|ELS| to |EULS|

=|NLS|$^2$*|ELS|, where NLS, ELS, and EULS stand for the label sets of nodes, edges, and edge units respectively.

3.2.2 Planning Model

Based on the six underlying concepts of *resource, task, constraint, solution, criticality and cruciality,* the presented approach treats SGI problem as being composed of EUs in $G_1$ to be assigned onto a set of EUs in $G_2$, under the condition that resources are constrained. The essential planning concepts when applied them onto SGI problem are defined below.

*Resource*: all unassigned EUs in $G_2$.
*Task(TEU)*: any unassigned EU in $G_1$. {Agenda: The set of unassigned EUs in $G_1$}.
*Solution(SEU)*: The EU in $G_2$ which is valid while
1) its edge label is identical to that of the TEU;
2) both nodes satisfy the degree and incident edge label set constraints with respect to that of the TEU. {Solution space: All valid EUs in $G_2$ which can be assigned by the TEU in $G_1$}.

*Constraints*:
1) Each TEU has to be assigned onto the SEU with identical edge label and node labels.
2) Node with degree $d_1$ and incident edge label set $LS_1$ in $G_1$ must be assigned to the node with degree $d_2$ and incident edge label set $LS_2$ in $G_2$ such that $LS_1 \subseteq LS_2$ and $d_1 \leq d_2$.

3.2.3 Operational Procedure

After preprocessing, the EUs in $G_2$ are treated as resource units, all TEUs in $G_1$ are considered as active tasks in agenda. Depending on the policy function (see 3.2.4), the agenda will be filtered out for TEU identification. Then, the set of valid SEUs will be scanned over for SEU selection according to another policy function (see 3.2.4). At each stage of iteration procedure, the topological connections between consistent TEUs and SEUs is verified such that the distance between *k*th TEU and *k+1*th TEU must be consistent to the distance between *k*th SEU and *k+1*th SEU, if there is an isomorphism. This is based on the path heuristic: Given two graphs $G_1$ and $G_2$, *if* $G_1$ *is isomorphic to* $G_2$ *then a* ***path*** *in* $G_1$ *must be isomorphic to a* ***path*** *in* $G_2$.

3.2.4 Policy Functions

The planning model provides two domain-independent guiding principles as the policy functions for TEU identification and SEU selection [12]. After estimating the number of possible, valid SEUs for each TEU, there are three possible cases of the estimation results. Case 1: Only *one* TEU can be assigned to exactly *one* SEU. In this case, it is nature to pick up the pair of TEU and SEU as a successful task allocation. This strategy is a natural generalization of a simple heuristic that says to always tackle the task with only one possible solution first (lest all needed resource units are consumed by performing other tasks and thereby reducing its own chance of completion). Case 2: *Multiple* TEUs have same number of valid SEUs. Guided by the ***most constrained strategy***, multiple TEUs are hierarchically evaluated by the ***criticality*** function [12] in order to identify the most constrained (*one*) TEU. The evaluation result might introduce the cases of either one TEU with one possible SEU which is case 1 or one TEU with multiple SEUs which is Case 3: *One* TEU has minimum (but *multiple*) number of valid SEUs. Then, the ***least impact strategy***, is used to direct the SEU selection for a chosen TEU. It selects the least impact SEU of the current TEU by minimizing the impact to other TEUs (measured hierarchically by the ***cruciality*** function [12], which calculates the reduction to the number of valid SEUs for other TEUs), thus maximizing the feasibility of completing all remaining TEUs. Both most constrained strategy and least impact strategy are composed of three hierarchical level calculations which are sequentially trying to conquer the conflicts on tasks, solutions, and resource units [12].

## 4. EXPERIMENTAL RESULTS

A C-implementation of the presented approach is carried out on Sun workstation 490. Np and Nm are node sets, Ep and Em are edge sets in pattern and model graphs, respectively. EULS is the label set on EUs and |EULS|=|NLS|$^2$*|ELS|, where NLS and ELS stand for the node label set and edge label set in model graph respectively. In practical cases for the special problem, the highest degree (h) in a graph is always bounded by a constant, such as h=3 for trehidral 3D object, h=6 for electronic circuit (gate), h=5 for protein sequence (almost tree). Therefore, a set of k-regular graphs (degree(v)=k, for all node v $\in$ N; N is node set of the graph) is generated for testing since a k-regular graph contains maximal number edges for the graph with highest degree k.

## 4.1 Efficiency

The performance of presented approach is shown in Figure 2. The execution time on matching among size-increasing pattern graphs and size-fixed model graph shows a near-linear curve trend with respect to the edge numbers of pattern graphs in the average cases of fixing the ratio of |EULS| and |Em|.

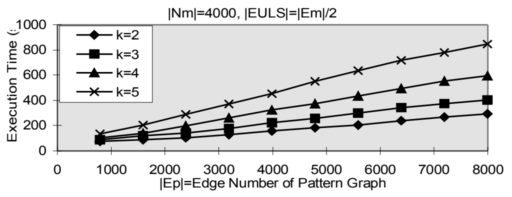

**Figure 2.** Near-Linear Performance

## 4.2 Impact of Label Sets

By testing a set of labeled graphs shows that ratio R=|EULS|/|Em| is critical to the performance (Figure 3). With our approach, if the ratio is roughly greater than 1/4, a near-constant execution time could be reached, i.e., for a majority of practical problem with reasonable amount of label information, the execution time using in solving SGI problem with our approach would merely depend on the edge numbers. This is another near-linear performance of the presented approach. And also, due to NP-completeness, exponential performance of unlabeled SGI is certainly the upper bound of SGI algorithms.

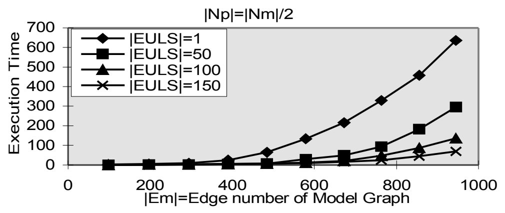

**Figure 3.** Impact of Label Sets

## 5. CONCLUSIONS

In this paper, a novel and efficient approach based on the resource management paradigm for subgraph isomorphism is presented. By integrating several powerful strategies, a near-linear performance is achieved for a wide range of pattern and model graph sizes. The subgraph isomorphism problem is usually applied to labeled graphs which contains the case of unlabeled graphs whose complexity is known to be NP-complete. The performance relationship derived for our SGI algorithm can be used to make a priori estimations of the expected computational costs in terms of the ratio of label set size and graph size. As a matter of fact, the demonstrated effectiveness of the new approach for solving the SGI problem can be applied to many application problem domains.

## REFERENCES

[1] P. J. Artymiuk, H. M. Grindly and etc., *"Identification of β -Sheet Motifs, of ψ -Loops, and of Patterns of Amino Acid Residues in Three-Dimensional Protein Structures Using a Subgraph-Isomorphism Algorithm"*, **J. Chem. Inf. Compute. Sci.** 1994, 34, 54-62

[2] D. G. Corneil and C. C. Gotlieb, *"An Efficient Algorithm for Graph Isomorphism"*, **J. Assoc. Compute.,** Vol. 17, No. 1, Jan. 1970, pp. 51-64.

[3] D. Eppstein, *"Subgraph Isomorphism in Planar Graphs and Related Problems",* Tech. Report 94-25, University of California, Irvine, May 31, 1994

[4] C. Ebeling and O. Zajicek, *"Validating VLSI Graph Layout by Wirelist Comparison",* Proc. of the Conference on Computer Aided Design (ICCAD), 172-173, 1983.

[5] M. R. Garey and D. S. Johnson., *"Computers and intractability: a guide to the theory of NP-completeness",* W. H. Freeman, 1979.

[6] N. P. Keng and D. Y. Y. Yun, "*A planning/Scheduling Methodology for the Constrained Resource Problem*", Proc. 1989 International Joint Conf. on Artificial Intelligence, pp. 20-25, Aug. 1989.

[7] Z. Ling and D. Y. Y. Yun, *"An efficient Subcircuit Extraction Algorithm by Resource Management Approach",* Proc. of Second International Conference On ASIC, Shanghai, P. R. China, 21-24 October 1996.

[8] M. Ohlrich, et al, "*SubGemini: Identifying SubGraphs using a Fast Subgraph Isomorphism Algorithm*", Proceeding of 30th ACM/IEEE Design Automation Conference, pp. 31-37, 1993.

[9] R. C. Read & D. G Corneil, "*The Graph Isomorphism Disease*". **J. of Graph Theory**, Vol. 1 (1977), pp. 339-363.

[10] J. R. Ullmann, *"An Algorithm for Subgraph Isomorphism"*, **J. Assoc. Compute.** March, 1976, 16, 31-42

[11] E. K. Wong, "*Model Matching in Robot Vision by Subgraph Isomorphism*", **Pattern Recognition**, Vol. 25. No. 3, pp. 287-303, 1992.

[12] D. Y. Y. Yun, H. M. Chen, Y. Q. Ge, and Z. Ling,, "*CRP-an Engine for Intelligent Resource Management*", to be submitted.