

A Fuzzy Attributed Graph Approach to Subcircuit Extraction Problem

Nian Zhang

Applied Computational Intelligence Lab
Dept. of Electrical and Computer Engineering
University of Missouri-Rolla
1870 Miner Circle
Rolla, MO 65409 USA
nzhang@umr.edu

Donald C. Wunsch II

Applied Computational Intelligence Lab
Dept. of Electrical and Computer Engineering
University of Missouri-Rolla
1870 Miner Circle
Rolla, MO 65409 USA
dwunsch@ece.umr.edu

Abstract- Fuzzy attributed graph (FAG) is not only widely used in the fields of image understanding and pattern recognition, but is useful to fuzzy graph matching problem. One of the applications of fuzzy graph matching is the subcircuit extraction problem. Subcircuit extraction problem is very important for VLSI testing, layout versus schematic (LVS) check, and circuit partition, etc. In this paper, fuzzy attributed graph (FAG) is first effectively applied to the subgraph isomorphism problem. And then we provide an efficient fuzzy attributed graph algorithm based on the solution to subgraph isomorphism for the subcircuit extraction problem. Similarity measurement makes a significant contribution to both the subgraph isomorphism problem and the subcircuit extraction problem.

1. INTRODUCTION

The subcircuit extraction is a very important application in many fields of computer aided design, especially for the present advanced VLSI design. For example, as VLSI design gets larger and larger every year, semiconductor industries have to ensure that there are no bugs in the design and that the manufactured chips are defect-free. Hence, it is desirable to identify the complete or partial set of invalid states before test generation [1]. In addition, subcircuit extraction is one of the operations in the layout vs. schematic (LVS) testing, which is a useful means of minimizing mask errors. Moreover, it is also an important application in circuit partition, and so on.

Several methods have been proposed to solve the subcircuit extraction problem. Yokomizo uses logic tree representation of a circuit. It is unsuitable for BiCMOS, bipolar and analog circuits [2]. SubGemini uses the partitioning and relabeling algorithm to recognize the subcircuits [3]. SUBGEN employs the partitioning and genetic algorithm to identify the subcircuits [4]. Ling [5] uses circuit decomposition and partition, as well as resource management paradigm to extract subcircuits. DECIDE algorithm [6] adopts a recursive scheme to achieve identification operation. It achieves faster speed than all the previous publications. In 2002, Wunsch and Zhang employed an adaptive critic design family -- heuristic dynamic programming (HDP) algorithm named SubHDP to solve the subcircuit extraction problem [7]. When compared with the DECIDE algorithm, the SubHDP algorithm can identify the NAND gate from the main circuit much faster when the main circuit has no more than 20,000 transistors. However, when the main circuit has more than 20,000 transistors, the DECIDE algorithm is faster. In this paper, we propose a novel fuzzy attributed graph approach to implement subcircuit extraction.

2. FUZZY ATTRIBUTED GRAPH

Attributed graph was introduced by Tsai and Fu for pattern analysis [8]. It gives a straightforward representation of structural patterns. The vertices of the graph represent pattern primitives describing the pattern while the arcs are the relations between these primitives. However, the pattern often possess properties that are fuzzy in nature and it has been extended to include fuzzy information into the attributes.

In a fuzzy attributed graph, each vertex may take attributes from the set $Z = \{z_i \mid i = 1, 2, \dots, I\}$ [9]. For each attribute z_i , it will take values from $S_i = \{s_{ij} \mid j = 1, 2, \dots, J_i\}$. The set of all possible fuzzy attribute-value pairs is $L_v = \{(z_i, A_{S_i}) \mid i = 1, \dots, I\}$, where A_{S_i} is a fuzzy set on the attribute-value set S_i . A valid pattern primitive is just a subset of L_v in which each attribute appears only once, and Π represent the set of all those valid pattern primitives. Thus, each vertex will be represented by an element of Π .

Similarly, each arc may take attributes from the set $F = \{f_i \mid i = 1, \dots, I\}$ in which each f_i may take values from $T_i = \{t_{ij} \mid j = 1, \dots, J_i\}$. And

$L_a = \{(f_i, B_{T_i}) \mid i = 1, \dots, I\}$ denotes the set of all possible relational attribute value pairs, where B_{T_i} is a fuzzy set on the relational attribute-value set T_i . A valid relation is just a subset of L_a in which each attribute appears only once. The set of all those valid relations is denoted as ϕ .

Definition 1: A Fuzzy Attributed Graph (FAG) over $L=(L_v, L_a)$, with an underlying graph structure $H=(N, E)$ is defined to be an ordered pair (V, A) , where $V=(N, \sigma)$ is called a fuzzy vertex set and $A=(E, \delta)$ is called a fuzzy arc set. The mapping $\sigma: N \rightarrow \Pi$ and $\delta: E \rightarrow \phi$ are called fuzzy vertex interpreter and fuzzy arc interpreter, respectively [10].

This definition also applies when there are non-fuzzy attributes, since a crisp (non-fuzzy) set can always be represented as a special case of a fuzzy set.

Definition 2: In pairing two fuzzy attributed graphs, nodes that are paired are named *core nodes*, nodes that are not paired but have branches directly connected to core nodes are named *goal nodes*, and the others are named *free nodes*.

3. FUZZY CIRCUIT GRAPH REPRESENTATION

3.1 Subgraph Isomorphism

The problem of subcircuit extraction can be transformed to the subgraph isomorphism problem. A graph G consists of a finite nonempty set $V = V(G)$ of p points together with a prescribed set X of q unordered pairs of distinct points of V [11]. Each pair $x = \{u, v\}$ of points in X is a line of G , and x is said to join u and v . A subgraph of G is a graph having all of its points and lines in G . The subgraph isomorphism problem is defined as: *Given a graph S and another larger graph T , to find all the subgraphs of T which are identified with S .* Similarly, the subcircuit extraction problem is to determine whether one given pattern circuit has any isomorphic subcircuits in another larger model circuit. A 2-input NAND gate serving as the subcircuit (i.e. pattern circuit) is shown in Fig. 1 (a). Its corresponding circuit graph is shown in Fig. 1 (b). In Fig. 1 (b), device is represented as a square, while terminal is represented as a circle. In Fig. 2, a netlist is shown as the main circuit, in which we will find the instances of the pattern circuit. Our goal is to verify that the netlist composed of M8, M9, M10 and M11 in the main circuit is isomorphic to the pattern circuit.

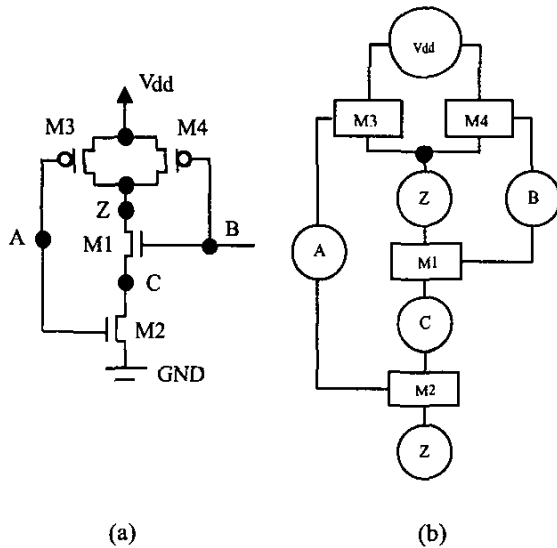


Fig. 1 (a) A CMOS circuit. It serves as the pattern circuit. (b) Corresponding graph representation of Fig. 1 (a).

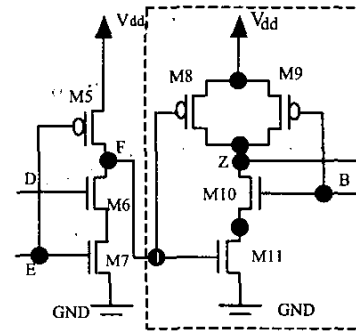


Fig. 2 Main Circuit

As we know, a circuit graph is a graph with electric elements and nodes as graph vertices. A circuit graph contains two types of vertices: device and terminal. The circuit graph is a bipartite graph, in which device vertices only connect to terminal vertices, and terminal vertices only connect to device vertices.

4. FUZZY ATTRIBUTED GRAPH MATCHING ALGORITHM ON SUBCIRCUIT EXTRACTION PROBLEM

4.1 Similarity of a Fuzzy Attributed Graph Pair

The definition of the similarity is adapted from [12]. We apply it to our subcircuit extraction problem. Assume we have a FAG pair: one is the pattern circuit graph, G_1 ; the other is a candidate subcircuit graph, G_2 . The i th vertex in G_1 and G_2 is denoted as n_x^i and n_y^i , respectively. Any node n_x^i and n_y^i can be represented by a vector of α_x^i and α_y^i , respectively. The edge between n_x^i and n_x^j in G_1 can be represented by a vector of $e_x^{(i,j)}$, and the edge between n_y^i and n_y^j in G_2 can be represented by a vector of $e_y^{(i,j)}$. Similarity between G_1 and G_2 is calculated with all the $\alpha_x^i, \alpha_y^i, e_x^{(i,j)}$ and $e_y^{(i,j)}$. The vertex in G_1 and G_2 can be expressed as $\alpha_x^i = (\alpha_x^i[1], \alpha_x^i[2], \alpha_x^i[3])^T$ and $\alpha_y^i = (\alpha_y^i[1], \alpha_y^i[2], \alpha_y^i[3])^T$, respectively, where $0 \leq \alpha_x^i[s], \alpha_y^i[s] \leq 1, (s = 1, 2, 3)$ are memberships that count how much an actual component belongs to a *N-type device*, a *P-type device* and a *terminal*, respectively. And $e_x^{(i,j)} = (\beta_x^{(i,j)})^T$, with $\beta_x^{(i,j)} = 0$ or 1 being membership that indicate

whether there is an edge between a device and a terminal.

If n_x^i is a core node in G1 and its pair node in G2 is n_y^j , a square distance between the two core nodes can be given by

$$DAC^i = \frac{\sum_{s=1}^3 (\alpha_x^i[s] - \alpha_y^j[s])^2}{3} \quad (1)$$

where DAC^i is a unified distance ($0 \leq DAC^i \leq 1$). For goal nodes and free nodes in G1 and G2, since there is no pair node, the corresponding square distances can be defined as distances to an zero vector as follows:

$$DAO_x^j = \frac{\sum_{s=1}^3 (\alpha_x^j[s])^2}{3} \quad (2)$$

$$DAO_y^j = \frac{\sum_{s=1}^3 (\alpha_y^j[s])^2}{3} \quad (3)$$

A synthesis square distance caused by all nodes then can be

$$DA = \frac{\sum_{i=1}^I DAC^i + \sum_{j_1=1}^{J_1} DAO_x^{j_1} + \sum_{j_2=1}^{J_2} DAO_y^{j_2}}{I + J_1 + J_2} \quad (4)$$

where I is the number of core nodes, J_1 the number of goal nodes and free nodes in G1, and J_2 the number of goal and free nodes in G2.

Similarly, if branches $e_x^{(i,j)}$ and $e_y^{(r_i,r_j)}$ are paired branches (i.e. nodes n_x^i and n_x^j are paired with $n_y^{r_i}$ and $n_y^{r_j}$, respectively), a square distance between two paired branches whose both end nodes are core nodes can be given as:

$$DEC^{(i,j)} = (\beta_x^{(i,j)} - \beta_y^{(r_i,r_j)})^2 \quad (5)$$

If $\beta_x^{(i,j)}$ and $\beta_y^{(r_i,r_j)}$ is the kth branch pair, then

$DEC^{(i,j)}$ can also be represented as DEC^k . Thus, (5) can also be written as:

$$DEC^k = (\beta_x^{(i,j)} - \beta_y^{(r_i,r_j)})^2$$

For other branches in G1 and G2, the corresponding square distances can be given by:

$$DEO_x^{l_1} = (\beta_x^{l_1})^2 \quad (6)$$

$$DEO_y^{l_2} = (\beta_y^{l_2})^2 \quad (7)$$

A synthesis square distance caused by all edges can also be given as:

$$DE = \frac{\sum_{k=1}^K DEC^k + \sum_{l_1=1}^{L_1} DEO_x^{l_1} + \sum_{l_2=1}^{L_2} DEO_y^{l_2}}{K + L_1 + L_2} \quad (8)$$

where K is the number of paired branches whose both end nodes are core nodes, L_1 the number of other branches in G1, and L_2 the number of other branches in G2.

It is clear that both DA and DE are unified. So a similarity measure between a FAG pair can be defined directly with

$$SIM(G1, G2) = 1 - \sqrt{\frac{w_a \cdot DA + w_e \cdot DE}{w_a + w_e}} \quad (9)$$

where w_a and w_e are properly selected weights, which are both set to 0.5 in our algorithm.

To determine the subgraph isomorphism between the two fuzzy attributed graphs, we should measure how much one FAG is a part of another. We denote the measurement as COM, shown in (10). Value '1' means a complete part.

$$COM = \max\{SIM(CORE_X(CN), G1), SIM(CORE_Y(CN), G2)\} \quad (10)$$

where CN is the number of core nodes, $CORE_X(CN)$ and $CORE_Y(CN)$ are two fuzzy attributed graphs consisting of only the core nodes in G1 and G2, respectively. This notation provides us a fast way to determine graph matching based on the percentage of the core nodes in each graph.

4.2 Fuzzy Graph Matching Algorithm on Subcircuit Extraction Problem

Given two fuzzy attributed graphs G1 and G2. G1 is the pattern circuit graph, and G2 is the candidate subcircuit graph. G1 has M nodes, which is represented as n_x^i ($i=1 \dots M$), and G2 has N nodes, which is represented as n_y^j ($j=1 \dots N$). Assume Vdd is the start node for both graphs. The algorithm is given below:

4.2.1. Circuit Setup

We partition the input circuit file into several netlists based on the characteristics of the input file [13]. If the integer array length of any netlist after the breadth-first search is the same as that of the pattern circuit, then it will be added to the candidate subcircuit list. And then we construct a graph for each candidate netlist by connecting devices to their three neighbors. In addition, it is important for us to create a Hash table because it has the following advantages:

1. We can easily get access to each vertex's information, such as type, flag and its neighbors.
2. It plays an indispensable role to find whether there is an edge between the core node and the element in the goal node set.
3. It is necessary to help to find out how many edges there are in the core node set.

The procedure to create Hash table is as follows: for a device, all of its three terminals will be added to its neighbor list; for a terminal, all of its connected devices will be added to its neighbor list. Thus, all the information of a device or a terminal is stored in a vector, which has a universal structure as [type flag neighbor 1 ... neighbor n].

4.2.2 Subcircuit Identification

Apparently, Vdd in G1 and Vdd in G2 is a core node pair. Thus, we first put Vdd of G1 into the core node set, CNA, and put Vdd of G2 into the core node set CNB. Set the flag attribute of Vdd to '1'. The following pseudo code is to implement the subcircuit identification for each candidate subcircuit.

```

OldCNA = 0; % a variable storing the length of the old CNA
while SIM ~ 1 % similarity measurement
    TA = [ ]; TB = [ ]; % set goal node sets to empty
    if length(CNA) == OldCNA % if the core node set
        didn't change
            break; % break the while loop;
        end
    Find out all the goal nodes corresponding to the current
    core node sets, CNA and CNB, respectively;
    if length(TA) ~= length(TB)
        Display error message, and then break;
    end
    for k = 1:length(TA)
        s(k) = 0; t(k) = 0; % s(k) means the number of
        branches from all the core nodes in CNA towards TA(k).
    Find out the number of edges from all the core nodes
    towards each element in the goal node set for both G1 and
    G2;
    end
    P = 1;
    OldCNA = length(CNA);
    for ss = 1:length(TA)
        for tt = P:length(TB)
            if s(ss) == t(tt)
                Add the goal node TA(ss) to the core node set, CNA,
                and add the goal node TB(tt) to CNB;
                Exchange TB(tt) and TB(P);
                P = P + 1;
                break;
            else
                if all the elements in TB has been explored
                    break the dual 'for' loops;
                end % if
            end % if
        end % for
    end % for
    if length(CNA) ~= length(CNB)
        Display error message and break;
    end
    Find out the number of paired branches between the two
    graphs whose both end nodes are core nodes;
    Calculate the similarity measurement;
    if SIM == 1 & length(CNB) == 10 % value '10' is the
    number of vertices in the pattern circuit (i.e. NAND gate).
    This condition ensures that all the nodes of both graphs are
    core nodes.
    Count this candidate subcircuit as an instance of the
    pattern circuit;
    break; % break the while loop
    end % if
end % while

```

The maximum SIM of the above process is the similarity between G1 and G2. The corresponding core node pairs represent the optimal matching result.

5. EXPERIMENTAL RESULTS

We use the Dell Precision Workstation 420, Intel Pentium III processors up to 800MHz, and 256MB memory to do the experiments. We try to identify the 2-input NAND gate from different size of RAM circuits.

We compare the total run time of our fuzzy attributed graph algorithm with the SubHDP algorithm, as shown in Fig. 3. The x-axis denotes the number of transistors in the main circuit, and the y-axis denotes the total run time. The red solid curve represents the SubHDP algorithm, and the blue dashed curve represents the fuzzy attributed graph algorithm, SubFuzzy. From Fig. 3, we find that: first, the SubHDP algorithm can identify the NAND gates from the main circuit faster in comparison with the SubFuzzy approach when the main circuit has no more than 15,000 transistors. However, when the main circuit has more than 15,000 transistors, our SubFuzzy algorithm is faster; second, the total run time of the SubFuzzy algorithm is approximately a smooth linear curve versus the number of transistors, while that of the SubHDP is a sharp curve with the main circuit size exceeding 50,000 transistors.

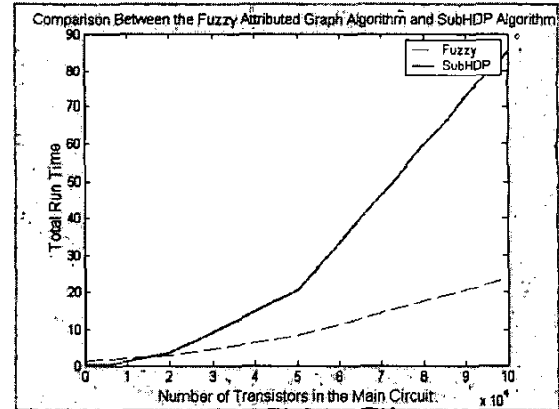


Fig. 3 Run time comparison between our fuzzy attributed graph algorithm and the SubHDP algorithm

6. CONCLUSION

From the experiments, we prove that the fuzzy attributed graph algorithm is an efficient approach to implement the subcircuit extraction problem. It can successfully find out all the instances of the pattern circuit (i.e. NAND gate) from the main circuit. In this paper, we use the NAND gate as our pattern circuit,

however, it can also be NOR gate, OAI gate, or RAM cell etc.

ACKNOWLEDGEMENT

The authors would like to acknowledge support of NSF and the MK Finley endowment.

[13] Nian Zhang, Frank Harary and Donald C. Wunsch II, "CMOS IC Topology Design Verification By Heuristic Dynamic Programming," *Proceedings of ANNIE '02*, St. Louis, MO, Nov.10 - 13, 2002.

REFERENCES

- [1] Michael John Sebastian Smith, "Application-Specific Integrated Circuits," Addison-Wesley, ISBN 0-201-50022-1, 1997.
- [2] G. Yokomizo, C. Yoshida, M. Miyama, M. Yousuke, and K. Nakajo, "A New Circuit Recognition Method for Pattern Based Circuit Simulation," *Proceeding of CICC*, May 1990.
- [3] Miles ohlrich, Carl Ebeling, Eka Ginting, and Lisa Sather, "SubGemini: Identifying Subcircuit using a Fast Subgraph Isomorphism Algorithm," *30th ACM/IEEE Design Automation Conference*, Dallas, Texas, June 14 - 18, 1993.
- [4] N. Vijaykrishnan and N. Ranganathan, "SUBGEN: A Genetic Approach for Subcircuit Extraction," *9th International Conference on VLSI Design*, January 1996.
- [5] Zong Ling and David Y.Y.Yun, "An Efficient Subcircuit Algorithm by Resource Management," *Proceeding of the Second International Conference of ASIC*, Shanghai, P.R.China, 1996.
- [6] Wei-Hsin Chang, Shuenn-Der Tzeng, and Chen-yi Lee, "A Novel Extraction Algorithm By Recursive Identification Scheme," *2001 IEEE International Symposium on Circuits and Systems*, Australia, 2001.
- [7] Nian Zhang and Donald C. Wunsch II, "A Novel Subcircuit Extraction Algorithm Using Heuristic Dynamic Programming (HDP)," *2002 International Conference on VLSI*, Las Vegas, Nevada, June 24 - 27, 2002.
- [8] Wen-Hsiang Tsai and King-Sun Fu, "Error-Correcting Isomorphisms of Attributed Relational Graphs for Pattern Analysis", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, pp. 757-768, 1979.
- [9] M.T.M.Gary and J.C.H Poon, "A Fuzzy-attributed Graph Approach to Handwritten Character Recognition," *Proceedings of the Second IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 570 - 575, 1993.
- [10] Kwok-Ping Chan, "Learning Templates from Fuzzy Examples in Structural Pattern Recognition," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 26, issue: 1, pp. 118 - 123, Feb. 1996.
- [11] Frank Harary, "Graph Theory," Narosa Publishing House, 1998.
- [12] Wei-Jie Liu and M. Sugeno, "A Similarity Measure of Fuzzy Attributed Graphs and its Application to Object Recognition," *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*, vol. 2, pp. 767 - 772, 1996.