JOURNAL OF ALGORITHMS **28,** 216–257 (1998)
ARTICLE NO. AL980942

The Recognition of Double Euler Trails in Series-Parallel Networks

Tung-Yang Ho

Department of Industrial Engineering and Management, Ta Hwa Institute of Technology, Hsinchu, Taiwan 30085, Republic of China

Ting-Yi Sung

Institute of Information Science, Academia Sinica, Taipei, Taiwan 11529, Republic of China

and

Lih-Hsing Hsu, Chang-Hsiung Tsai, and Jeng-Yan Hwang

Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 30050, Republic of China

Received April 20, 1994; revised March 31, 1998

Given a series-parallel network (network, for short) N, its dual network N' is given by interchanging the series connection and the parallel connection of network N. We usually use a series-parallel graph to represent a network. Let $G[N]$ and $G[N']$ be graph representations of N and N', respectively. A sequence of edges $e_1, e_2, \ldots, e_k$ is said to form a common trail on $(G[N], G[N'])$ if it is a trail on both $G[N]$ and $G[N']$. If a common trail covers all of the edges in $G[N]$ and $G[N']$, it is called a *double Euler trail*. However, there are many different graph representations for a network. We say that a network N has a double Euler trail (DET) if there is a common Euler trail for some $G[N]$ and some $G[N']$. Finding a DET in a network is essential for optimizing the layout area of a complementary CMOS functional cell. Maziasz and Hayes (*IEEE Trans. Computer-Aided Design* **9** (1990), 708–719) gave a linear time algorithm for solving the layout problem in fixed $G[N]$ and $G[N']$ and an exponential algorithm for finding the optimal cover in a network without fixing graph representations. In this paper, we study properties of subnetworks of a DET network. According to these properties, we propose an algorithm that automatically generates the rules for composition of trail cover classes. On the basis of these rules, a linear time algorithm for recognizing DET networks is presented. Furthermore, we also give a necessary and sufficient condition for the existence of a double Euler circuit in a network.

0196-6774/98 $25.00

1. INTRODUCTION

A *series-parallel network* (*network*, for short) N of *type* $t \in \{L, S, P\}$ (which represent *leaf*, *series*, and *parallel*, respectively) defined on W is recursively constructed as follows:

(i) N is a network of type L if $|W| = 1$.

(ii) If $|W| > 1$, N is a network of either type P or type S and consists of $k \geq 2$ networks $N_1, \ldots, N_k$ as *child subnetworks* parallel or series connected together, where each N_i is defined on a set W_i of type t_i with $t_i \neq t$ and the collection of W_i's forms a partition of W.

A network is often expressed by a tree structure. Networks are useful in practice since they correspond to Boolean formulas with series connection (denoted by S) implementing logical-AND and parallel connection (denoted by P) implementing logical-OR. For example, the Boolean function $\mathbf{e} \wedge (\mathbf{a} \vee \mathbf{b}) \wedge (\mathbf{c} \vee \mathbf{d})$ can be represented by the network shown in Figure 1. Moreover, networks can be used as a model for electrical circuits. For example, we can use the tree structure shown in Figure 1 to represent the network corresponding to the electrical circuit shown in Figure 2. In the tree representation of N, every node together with all of its descendants forms a *subnetwork* of N. A node together with some children and their descendants forms a *partial subnetwork*. The subnetwork of N formed by a child of the root is called a *child subnetwork* of N. The leaf node is labeled by x if it is a subnetwork of type L defined on $\{x\}$. Every *internal node* is labeled by S or P according to the type of the subnetwork it represents. Note that the order of the subtrees in the tree representation is immaterial, because different orders lead to the same Boolean formula.

On the other hand, every network can be represented by a *series-parallel graph* (*s.p. graph* for short), which is an edge-labeled graph with two given distinguished vertices denoted by s and t. We recursively construct an s.p.

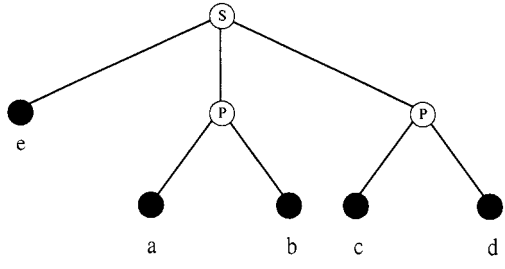


FIG. 1. Tree representation of a series-parallel network.

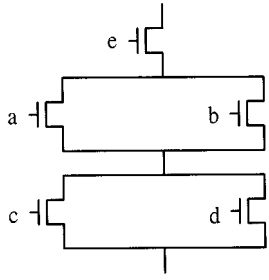


FIG. 2. Electrical circuit corresponding to the network shown in Figure 1.

graph to represent a network as described below:

(i) Every network N defined on $W = \{x\}$ of type L is represented by an edge-labeled graph $G[N]$ having only one edge labeled x and the two end points of this edge as distinguished vertices.

(ii) Let N be a network having child subnetworks $N_1, \ldots, N_k$, and let $G[N_i]$ be an s.p. graph representing N_i with the distinguished vertices s_i, t_i for every i. For N of type S, we identify t_i with s_{i+1} for $1 \leq i \leq k - 1$. The resulting graph $G[N]$ with the distinguished vertices s_1, t_k represents the network N. For N of type P, we identify all s_i's to obtain a new vertex s and identify all t_i's to obtain a new vertex t. The resulting graph $G[N]$ with distinguished vertices s, t represents the network N.

The subgraph G_i of G induced by the child subnetwork N_i of N is called a *child s.p. subgraph* of G. We note that a graph representation for a network is not unique because we can vary the order of the subnetworks and the order of the two distinguished vertices to obtain different graph representations. For example, both the nonisomorphic s.p. graphs shown in Figure 3 represent the network in Figure 1. Although most research has concentrated on s.p. graphs [2, 5, 8, 13, 14, 16] rather than on networks [6, 7, 9] we believe that studying networks is interesting and practical, although difficult.

Given a network N on set W, we define its *dual network* N' on set W by interchanging the types S or P of each node. For example, the network in Figure 4a is the dual network of the network in Figure 1. Figures 4b and 4c show a graph representation and the corresponding circuit, respectively, of the dual network. Note that the Boolean formula corresponding to N' is the dual of the Boolean formula that corresponds to N. It is obvious that $(N')' = N$. If two s.p. graphs G, G' represent some network N and its dual, respectively, we say that (G, G') is an *s.p. graph pair*.

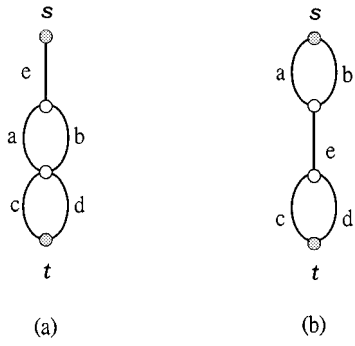


FIG. 3. Two nonisomorphic s.p. graphs representing the network in Figure 1.

A *walk* $W = v_0, e_1, v_1, e_2, \ldots, e_k, v_k$ is a finite non-null sequence of vertices and edges where $e_i = (v_{i-1}, v_i)$ for $1 \leq i \leq k$. Furthermore, we call W a walk from v_0 to v_k or a (v_0, v_k)-walk. The vertices v_0 and v_k are called *terminals* of W, and $v_1, \ldots, v_{k-1}$ are called *internal vertices*. We sometimes express a walk $v_0, e_1, v_1, e_2, \ldots, e_k, v_k$ as $e_1, e_2, \ldots, e_k$ for convenience. A *section* of a walk W is a walk that is a subsequence

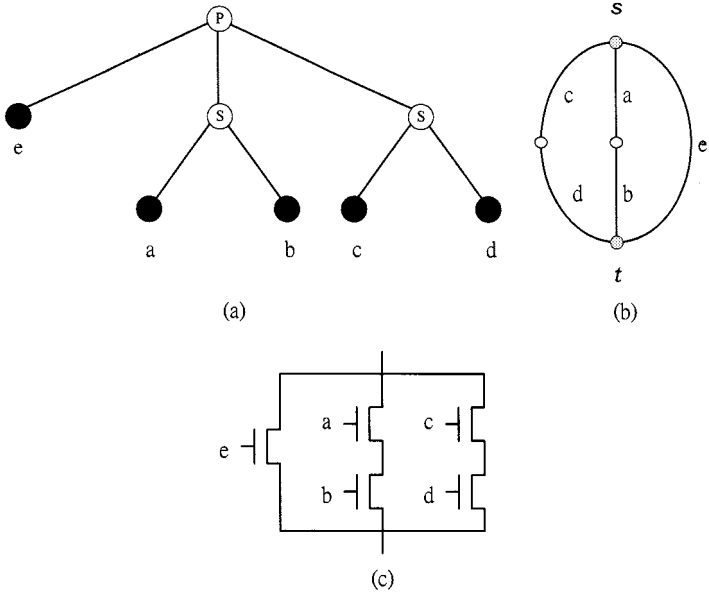


FIG. 4. (a) The dual network of the network in Figure 1. (b) A graph representation corresponding to the network in (a). (c) Electrical circuit corresponding to the network in (a).

$v_i, e_{i+1}, v_{i+1}, e_{i+2}, \ldots, e_j, v_j$ in W. If all edges of walk W are distinct, W is called a *trail*.

Given a network N, let $G[N]$ and $G[N']$ be graph representations for the network N and its dual network N', respectively. A sequence of edges $e_1, \ldots, e_m$ is said to form a *common trail* in $(G[N], G[N'])$ if $L = v_0, e_1, v_1, e_2, \ldots, e_m, v_m$ and $L' = v'_0, e_1, v'_1, e_2, \ldots, e_m, v'_m$ are trails in $G[N]$ and $G[N']$, respectively. We call (L, L') a trail pair. (We always use L' to denote a trail for the dual network.) We sometimes write (L, L') as L for short. If $v_0 = v_m$ and $v'_0 = v'_m$, we say that $e_1, \ldots, e_m$ form a *common circuit*. Furthermore, if $e_1, e_2, \ldots, e_m$ are all of the edges in both $G[N]$ and $G[N']$, we say that $e_1, \ldots, e_m$ form a *common Euler trail* (*circuit*) in $(G[N], G[N'])$. A network N has a *double Euler circuit* (DEC) if there is a common Euler circuit for both some $G[N]$ of N and some $G[N']$ of the dual network N'. A network N has a *double Euler trail* (DET) if there is a common Euler trail for both some $G[N]$ of N and some $G[N']$ of the dual network N'. We say that $(G[N], G[N'])$ *realizes a DET pair* (L, L') for N and N', where L and L' are the corresponding Euler trails in $G[N]$ and $G[N']$, respectively. We say that a network N is DET if N possesses a DET. For example, the s.p. graphs in Figures 3a and 4b do not have a common Euler trail, but the ones in Figures 3b and 4b have a common Euler trail **abecd**. Thus the network in Figure 1 is DET.

The problem of DET networks arises from a more general problem called $DCT(N)$. Let $DCT(G[N], G[N'])$ be the minimum number of disjoint common trails that cover all of the edges in $G[N]$ and $G[N']$. We define $DCT(N)$ as the minimum of $DCT(G[N], G[N'])$ among all possible graph representations $G[N]$ and $G[N']$. Uehara and vanCleemput [15] proposed a solution method for the layout of cells in the style shown in Figure 5. Assuming the height of each cell is fixed by technological considerations, the width of the cell, and therefore the area of the cell, can be minimized by ordering the transistors in the layout so that chains of transistors can share a common diffusion region. Uehara and vanCleemput defined a graph model for functional cells on two dual multigraphs $(G[N], G[N'])$ and proposed a heuristic method for finding a small number of common trails that cover the given $(G[N], G[N'])$. Maziasz and Hayes [11] gave a linear time algorithm for solving $DCT(G[N], G[N'])$ and an exponential algorithm for finding the $DCT(N)$. Several other papers have also explored the use of graph models to find solutions for layout [4, 10, 11, 12]. However, using this approach, the choice of graph model becomes a critical issue. We thus choose to work on networks instead. Solving $DCT(N)$ is useful but difficult. Therefore, we start by solving the DET problem on networks with the hope of solving $DCT(N)$ in the future, since $DCT(N) = 1$ if and only if N is DET.

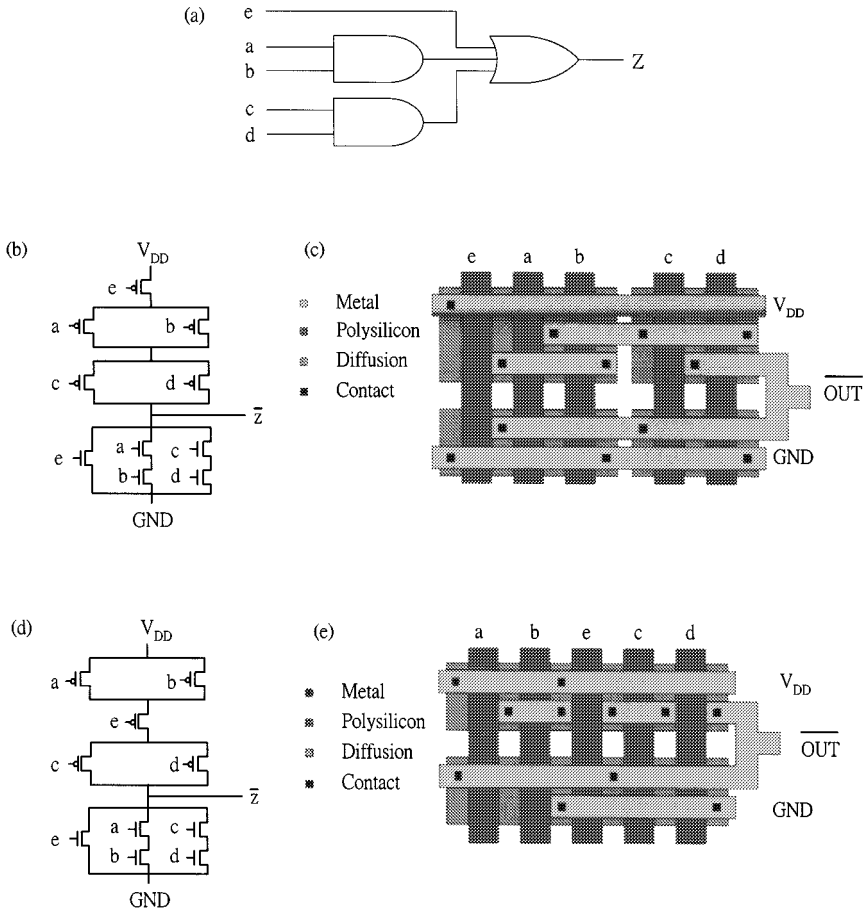


FIG. 5. CMOS functional cell. (a) Gate-level scheme. (b) An electric-level scheme. (c) Geometric layout corresponding to the scheme in (b). (d) Another electric-level scheme. (e) Geometric layout corresponding to the scheme in (d).

In this paper we study the properties of DET networks and give a linear time algorithm for recognizing DET networks. The paper is organized as follows. In Section 2, we classify the trail cover classes. In Section 3, we study properties of subnetworks of DET networks. On the basis of the analysis in Section 3, we present an algorithm in Section 4 to generate the rules for trail cover class composition. Using these rules, we give a necessary and sufficient condition for DEC networks in Section 5. A linear time algorithm for recognizing DET networks is also presented in Section

5. An example for illustration of the algorithm is given in Section 6. Finally, we give concluding remarks in Section 7.

2. COMMON TRAIL COVER CLASSES AND NETWORK CLASSES

We first informally use an example to introduce our terminology. We illustrate in Figure 6 a graph representation $(G[N], G[N'])$ for a DET network N shown in Figure 7. (At this moment, Figure 7 is used only for its tree representation for network N.) In this graph pair, there are 10 child s.p. subgraph pairs (G_i, G_i'), each corresponding to a child subnetwork N_i of N. $N_1, N_2, \ldots, N_{10}$ are ordered from left to right of the root node in Figure 7. The trail $L = 1, 2, \ldots, 46$ is a DET trail in $(G[N], G[N'])$. Let the distinguished vertices of G_8 be s_8 and t_8, where s_8 is on the top of G_8. Let the corresponding distinguished vertices of G_8' be denoted by s' and t'. The trail L induces two disjoint common trails $L_1 = 31, 32$ and $L_2 = 39, 40, \ldots, 46$ that cover all of the edges in (G_8, G_8'). We call $\{L_1, L_2\}$ the *trail cover* induced by (L, L') in (G_8, G_8'). On the other hand, (L, L') can be treated as the concatenation of all of the disjoint trail covers induced by (L, L') in G_i for $i = 1, 2, \ldots, 10$. To define trail cover types, we first need to define trail types. For example, the trail L_1 in G_8 begins at s_8 and terminates at t_8, whereas L_1' begins at s' and terminates at s'. We say that the trail type of (L_1, L_1') in (G_8, G_8') is $(s, s)/(t, s)$, where $(x, y)/(z, w) =$ (the beginning vertex of L_1, the beginning vertex of L_1')/(the ending vertex of L_1, the ending vertex of L_1'). All of the subscripts and superscripts are omitted for simplification. For the same reason, the trail L_2 in G_8 begins at t and terminates at s, whereas L_2' in G_8' begins at t' and terminates at an internal vertex of G_8'. We say that the trail type (L_2, L_2') in (G_8, G_8') is $(t, t)/(s, I)$, where I denotes an internal vertex. Since the terminal vertex of L_2' is an internal vertex, L_2 can only concatenate with another trail from its beginning vertex. For this reason, we change the trail type of L_2 into $(t, t)/(I, I)$. The trail cover type of $\{L_1, L_2\}$ in G_8 is then determined by the trail types of L_1 and L_2 and is given by $\{(s, s)/(t, s) + (t, t)/(I, I)\}$.

Now we formally define the term *trail cover type*. Let N be a network with graph representation $(G[N], G[N'])$. Let $\mathscr{L}$ be a family of disjoint common trails that cover all of the edges in $(G[N], G[N'])$. Let (L, L') be a trail pair in $\mathscr{L}$ with v_0, v_n and v_0', v_n' as terminals in L and L', respectively. If the terminals of both L and L' are distinguished vertices of $(G[N], G[N'])$, (L, L') is called a *distinguished trail*, and a *nondistinguished trail* otherwise. To be specific, the *type* of (L, L') in $(G[N], G[N'])$, denoted by $T(L, L')$, has the form $(T(v_0), T(v_0'))/(T(v_n), T(v_n'))$, where $T(v)$ can be a distinguished vertex (s or t) or an internal vertex (denoted

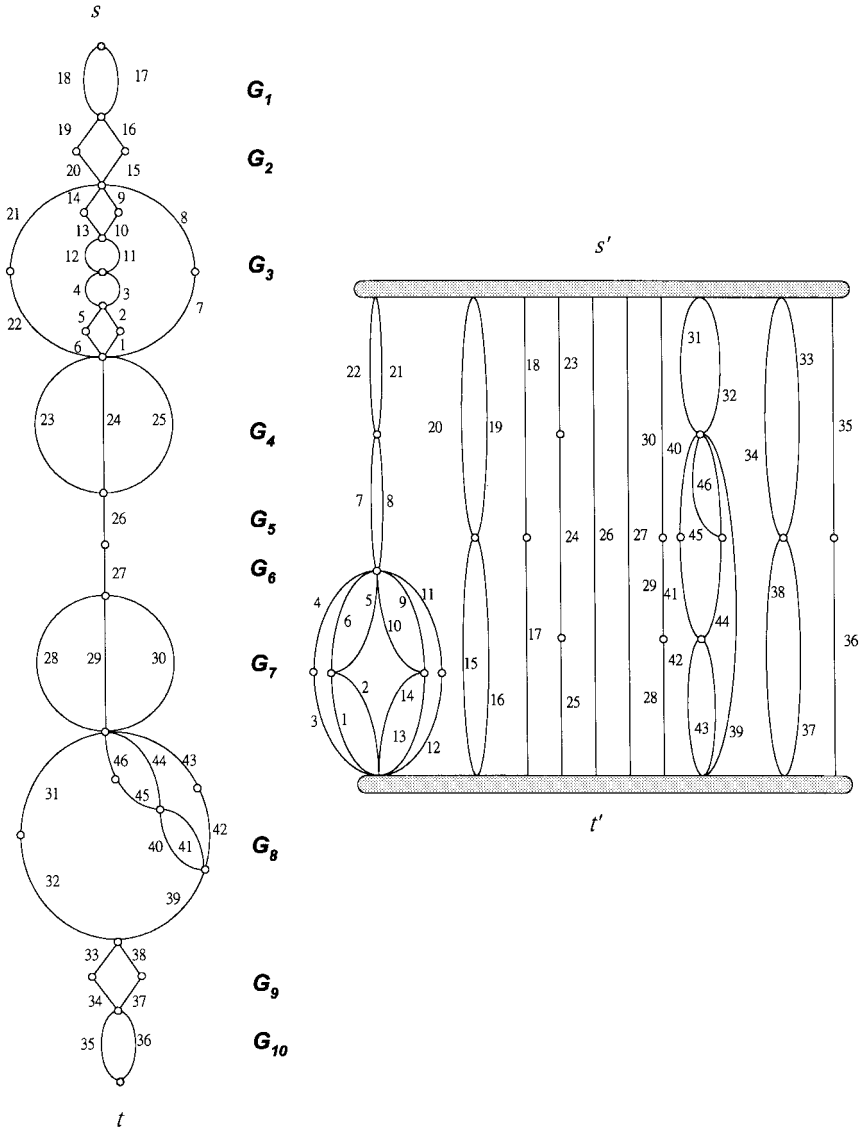


FIG. 6. An s.p. graph pair with a DET trail.

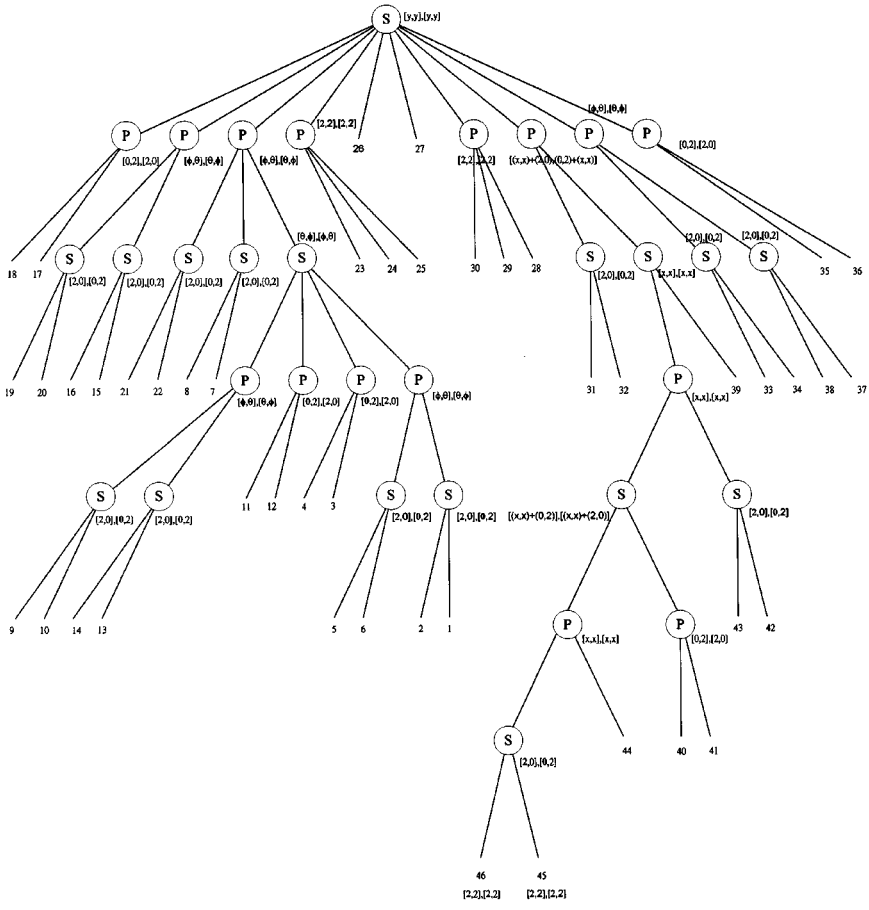


FIG. 7. The tree representation of the network with an s.p. graph pair shown in Figure 6.

by I). The order of the terminals in a trail type is irrelevant; e.g., $(t, t)/(I, I)$ and $(I, I)/(t, t)$ are considered to be equivalent. A nondistinguished trail can concatenate with at most one other trail. Since we are considering the concatenation of common trails, types $\{(t, I), (s, I), (I, t), (I, s)\}$ of $(T(v_0), T(v_0'))$ or $(T(v_n), T(v_n'))$ can be represented by (I, I); e.g., $(t, s)/(t, I)$ is denoted by $(t, s)/(I, I)$, and $(I, t)/(t, I)$ is denoted by $(I, I)/(I, I)$.

Let $\mathscr{L} = \{(L_1, L_1'), (L_2, L_2'), \ldots, (L_k, L_k')\}$ be a set of disjoint common trails that cover all of the edges in $(G[N], G[N'])$ for some network N. The *trail cover type* $\tau(\mathscr{L})$ is defined as $\tau(\mathscr{L}) = \{T(L_1, L_1') + T(L_2, L_2') + \cdots + T(L_k, L_k')\}$. Throughout this paper, "+" is commutative. We define *dual trail cover type* for $\tau(\mathscr{L})$, i.e., a trail cover type for $\mathscr{L}'$ in $(G[N'], G[N])$ as $\tau'(\mathscr{L}) = \{T(L_1', L_1) + T(L_2', L_2) + \cdots + T(L_k', L_k)\}$.

Let M be a network with graph representation $(G[M], G[M'])$ and $\mathscr{L}$ be a set of disjoint common trails that cover all of the edges in $(G[M], G[M'])$. For a partial subnetwork R of M, we denote by $G_M[R]$ the subgraph of $G[M]$ induced by R. Let N be a partial subnetwork of M such that $G_M[N]$ and $G_{M'}[N']$ are connected subgraphs of $G[M]$ and $G[M']$, respectively. The edges of $(G_M[N], G_{M'}[N'])$ constitute a set of maximal sections of $\mathscr{L}$ that cover edges of $(G_M[N], G_{M'}[N'])$. Let $\tau_G(N, \mathscr{L})$ denote the trail cover type on $(G_M[N], G_{M'}[N'])$ derived from $\mathscr{L}$. To classify different types in $\tau_G(N, \mathscr{L})$, we note that any common trail is constructed from $\{(t,t)/(s,s); (t,s)/(s,t)\}$, which represents the trail cover type of a leaf, by a sequence of series connection, parallel connection, and taking the duals. Maziasz and Hayes [10, 11] considered the closure of all series connection, parallel connection, and taking the duals of the trail covers generated by $\{(t,t)/(s,s); (t,s)/(s,t)\}$. They classified $\tau_G(N, \mathscr{L})$ into 42 types according to the directions of each trail. In their classification, there is no common trail that begins and ends at the same distinguished vertex in both $G[N]$ and $G[N']$, e.g., $(t,t)/(t,t)$. Furthermore, nondistinguished trails are represented by $(t,t)/(I,I)$, $(t,s)/(I,I)$, $(s,t)/(I,I)$, $(s,s)/(I,I)$, and $(I,I)/(I,I)$.

To simplify the exposition of this 42-type classification, we define an equivalence relation of trail cover types, which renders the concept of *trail cover class*. We use the example of Figure 6 to informally introduce this concept. A network may have different graph representations. Given a graph representation of a network, we can obtain other graph representations by a sequence of interchanging the distinguished vertices of their s.p. subgraphs. For this reason, the trail type $(s,s)/(t,s)$ of (L_1, L_1') in (G_8, G_8') in Figure 6 may change into $(s,t)/(t,s)$, $(t,s)/(s,s)$, or $(t,t)/(s,t)$. All of these types represent a common trail (L_1, L_1') in which L_1 begins at a distinguished vertex and terminates at the other distinguished vertex, whereas L_1' begins at one distinguished vertex and terminates at the same vertex. It is observed that the subgraph of G_8 induced by 31, 32 has exactly two vertices of odd degree, namely, the distinguished vertices of G_8. We use 2 to indicate the trail class of L_1. Similarly, we use 0 to indicate the trail class of L_1' because none of its distinguished vertices are of odd degree. We say that the trail class of (L_1, L_1') in (G_8, G_8') is $[2, 0]$. We use x to indicate a trail in G in which one end point is a distinguished vertex of G and the other an internal vertex, and y to indicate a trail in G in which L begins and terminates at internal vertices. In our example, the trail class of (L_2, L_2') is $[2, x]$. However, since $[2, x]$ can concatenate with another common trail only at one end, we change trail class $[2, x]$ into $[x, x]$. In other words, as $(t,s)/(t,I)$ is represented by $(t,s)/(I,I)$, we can write the trail class of (L_2, L_2') as $[x, x]$ instead of $[2, x]$. Since $\{L_1, L_2\}$ forms a trail cover of G_8, we say the trail cover class of $\{L_1, L_2\}$ is $[(x, x) + (2, 0)]$.

To formally define the term *trail cover class*, we first define an equivalence relation of trail cover types as follows. For a graph representation $(G[M], G[M'])$ of a network M, a trail cover type τ in $(G[M], G[M'])$ is equivalent to $\bar{\tau}$ if and only if τ can be obtained from $\bar{\tau}$ by permuting the distinguished vertices of $\bar{\tau}$ in $G[M]$ or $G[M']$ or both, i.e., by reversing the direction of $\bar{\tau}$ in $G[M]$ or $G[M']$ or both. This equivalence relation enables us to define a trail cover class that induces all trail cover types that are equivalent to each other. To consider the equivalence of trail cover types, we need to compare the trail sets in primal and dual graphs, respectively. Now we are motivated to introduce new definitions of trail classes in an s.p. graph without considering the direction of a trail.

DEFINITION 2.1. Given an s.p. graph G with distinguished vertices s and t, let L be a trail of G. (Note that this trail does not necessarily contain all of the edges of G.) We say that L is in class 2 if L begins at a distinguished vertex and terminates at the other distinguished vertex, in class 0 if L begins at a distinguished vertex and returns to the same distinguished vertex, in class x if one end point of L is a distinguished vertex and the other is an internal vertex, and in class y if L begins and terminates at internal vertices.

As shown in Figure 6, the trail $(23, 24, 25)$ of G_4 is in class 2, the trail $(17, 18)$ of G_1 in class 0, the trail $(40, 41, \ldots, 46)$ of G_8 in class x, and the trail $(1, 2, \ldots, 46)$ of G in class y.

Given an s.p. graph G, let $T = \{L_1, L_2\}$ be a set of two disjoint trails in G. We say that T is in class θ if both L_1 and L_2 are in class 0 and begin and end at different distinguished vertices. We use θ to symbolize a trail cover consisting of two class 0 trails as top and bottom parts. If L_1 and L_2 are in class 2 and begin and end at the same distinguished vertex, T is said to be in class ϕ, which is used to symbolize the left and right parts of T. Similarly, we define classes θx and ϕx for the case where L_1 and L_2 are in class x. We say that T is in class θx if L_1 and L_2 begin or terminate at different distinguished vertices, and in class ϕx if L_1 and L_2 begin or terminate at the same distinguished vertex. Suppose that if L_1 is in class 0 and L_2 is in class x, we say that T is in class $0 + x$. Other trail cover classes can be similarly defined for trail sets with more than two trails of other combinations.

Again we use Figure 6 as an example. We can consider a trail set in G_i, which does not necessarily contain all of the edges in G_i, but contains all of the edges in a subgraph of G_i that corresponds to a partial subnetwork. The trail set $\{(15, 16), (19, 20)\}$ in G_2 is in class ϕ, the trail set $\{(1, 2, \ldots, 6), (9, 10, \ldots, 14)\}$ in G_3 is in class θ. In the example of Figure 6, we cannot find a trail set induced by L which is in class θx or ϕx. Since a graph can have many different trail sets, for illustration purposes we define other

trail sets in G_8 and G_3. New trail sets in G_8 and G_3 are given by $\{(41, 40, 39), (42, 43, 44, 45, 46)\}$, and $\{(3, 2, 1, 7, 8), (11, 10, 9, 21, 22)\}$, respectively. The trail set $\{(41, 40, 39), (42, 43, 44, 45, 46)\}$ in G_8 is in class θx, and $\{(3, 2, 1, 7, 8), (11, 10, 9, 21, 22)\}$ in class ϕx.

Let M be a network that realizes a DET (L, L') in $(G[M], G[M'])$. For a partial subnetwork R of M, L induces a trail cover in $(G_M[R], G_{M'}[R'])$, which is denoted by $L_G[R]$. Using the notation of trail classes 0, 2, x, y, θ, ϕ, θx, and ϕx, we can reclassify the 42 trail cover types proposed by Maziasz and Hayes [10, 11] into the following 18 trail cover classes:

class a: $[2, 2] = \{(t, t)/(s, s); (t, s)/(s, t)\}$

class b: $[2, 0] = \{(t, t)/(s, t); (t, s)/(s, s)\}$

class c: $[0, 2] = \{(t, t)/(t, s); (s, t)/(s, s)\}$

class d: $[\theta, \phi] = \{(t, t)/(t, s) + (s, t)/(s, s)\}$

class e: $[\phi, \theta] = \{(t, t)/(s, t) + (t, s)/(s, s)\}$

class f: $[x, x] = \{(t, t)/(I, I); (t, s)/(I, I); (s, t)/(I, I); (s, s)/(I, I)\}$

class g: $[(x, x) + (2, 2)] = \{(t, t)/(s, s) + (t, s)/(I, I);\ (t, t)/(s, s) + (s, t)/(I, I); (t, s)/(s, t) + (t, t)/(I, I); (t, s)/(s, t) + (s, s)/(I, I)\}$

class h: $[(x, x) + (0, 2)] = \{(t, t)/(t, s) + (s, t)/(I, I);\ (t, t)/(t, s) + (s, s)/(I, I); (s, t)/(s, s) + (t, t)/(I, I); (s, t)/(s, s) + (t, s)/(I, I)\}$

class i: $[(x, x) + (2, 0)] = \{(t, t)/(s, t) + (t, s)/(I, I);\ (t, t)/(s, t) + (s, s)/(I, I); (t, s)/(s, s) + (t, t)/(I, I); (t, s)/(s, s) + (s, t)/(I, I)\}$

class j: $[\phi x, \theta x] = \{(t, t)/(I, I) + (t, s)/(I, I);\ (s, t)/(I, I) + (s, s)/(I, I)\}$

class k: $[\theta x, \phi x] = \{(t, t)/(I, I) + (s, t)/(I, I);\ (t, s)/(I, I) + (s, s)/(I, I)\}$

class l: $[\theta x, \theta x] = \{(t, t)/(I, I) + (s, s)/(I, I); (t, s)/(I, I) + (s, t)/(I, I)\}$

class m: $[(\theta x, \theta x) + (2, 2)] = \{(t, s)/(s, t) + (t, t)/(I, I) + (s, s)/(I, I); (t, t)/(s, s) + (t, s)/(I, I) + (s, t)/(I, I)\}$

class n: $[(\phi x, \theta x) + (0, 2)] = \{(t, t)/(t, s) + (s, t)/(I, I) + (s, s)/(I, I); (s, t)/(s, s) + (t, t)/(I, I) + (t, s)/(I, I)\}$

class o: $[(\theta x, \phi x) + (2, 0)] = \{(t, t)/(s, t) + (t, s)/(I, I) + (s, s)/(I, I); (t, s)/(s, s) + (t, t)/(I, I) + (s, t)/(I, I)\}$

class p: $[y, y] = \{(I, I)/(I, I)\}$

class q: $[3(x, x)] = \{(t, t)/(I, I) + (t, s)/(I, I) + (s, t)/(I, I); (t, t)/(I, I) + (t, s)/(I, I) + (s, s)/(I, I);\ (t, t)/(I, I) + (s, t)/(I, I) + (s, s)/(I, I); (t, s)/(I, I) + (s, t)/(I, I) + (s, s)/(I, I)\}$

class r: $[4(x, x)] = \{(t, t)/(I, I) + (t, s)/(I, I) + (s, t)/(I, I) + (s, s)/(I, I)\}$.

Theorem 1 of [11] states that these 42 trail cover types define a complete set of trail cover types. Since the nondistinguished trail of type $(I, I)/(I, I)$ cannot be concatenated with any other trail, it stands in class p itself. Since the nondistinguished trail with one distinguished vertex as a terminal in each trail cover can concatenate with at most one other trail, it must be the beginning or ending section of the trail cover. Therefore, no DET network can have trail covers that include more than two of such nondistinguished trails. In other words, no DET network can have trail covers in classes q and r. Thus we eliminate these two classes from our analysis and focus on the first 16 trail cover classes, $a, \ldots, p$ only. We restrict all of the trail covers in the following discussion to be in a subset of {class a, class $b, \ldots,$ class p}. For simplicity, we write $\{a, b, \ldots, p\}$.

It is necessary to distinguish trail cover classes on series-type networks from those on parallel-type networks. The dual class of a trail cover class, say $[w, z]_S$ ($[z, w]_P$) in (N, N'), is defined as $[z, w]_P$ ($[w, z]_S$) in (N', N). The dual class is obtained by reversing the role of primal and dual networks. Since the type S or P of a network is given, we can sometimes omit the subscripts S or P of a trail cover class without ambiguity. We call $[z, w]$ the dual trail cover class (or simply, dual class) of $[w, z]$. For example, the dual class of $[(\phi x, \theta x) + (0, 2)]$ is $[(\theta x, \phi x) + (2, 0)]$, since the dual of $[\phi x, \theta x]$ is $[\theta x, \phi x]$ and the dual of $[0, 2]$ is $[2, 0]$. In Figure 7 we also show the trail cover class and the dual trail cover class associated with each node derived from $L = 1, 2, \ldots, 46$. Note that the trail cover class and the dual trail cover class notations in Figure 7 are for the particular graph shown in Figure 6 and the specific DET L. It is possible to obtain other trail cover classes by rearrangement of subgraphs, i.e., different concatenation of s.p. graphs. In other words, a different concatenation of s.p. graphs can yield a different concatenation of trail cover classes. (Later in Lemma 4.1 and Table 2 of Section 4, we will show that some specific concatenations of s.p. graphs or trail cover classes are preferred to obtain a DET.)

We define a different concatenation of s.p. graphs as follows. Let G_1 and G_2 be two s.p. graphs having distinguished vertices s_k and t_k for $k = 1, 2$. (In this paper, we use s to denote the distinguished vertex on the top of an s.p. graph and t to denote the one on the bottom.) When s_k and t_k, $k = 1, 2$, are specified, we can define a different concatenation of G_1 and G_2. We use $G_1 \sigma^i G_2$ and $G_2 \sigma^i G_1$ to denote all possible series connections of G_1 and G_2, where $G_{j_1} \sigma^i G_{j_2}$ represents an s.p. graph with G_{j_1} placed on top of G_{j_2} for $j_1, j_2 \in \{1, 2\}$ and $j_1 \neq j_2$. To be specific, $G_1 \sigma^1 G_2$ denotes the resulting s.p. graph with distinguished vertices $s = s_1$ and $t = t_2$. The remaining three concatenations of $G_1 \sigma^i G_2$ for $i = 2, 3, 4$ can be similarly defined. Furthermore, we can similarly define $G_2 \sigma^i G_1$ for

$i = 1, 2, 3, 4$, and parallel connections $G_1\pi^i G_2$, $G_2\pi^i G_1$ as well. We use $G_1\sigma G_2$ to denote the set $\{G_1\sigma^i G_2 \mid i = 1, 2, 3, 4\} \cup \{G_2\sigma^i G_1 \mid i = 1, 2, 3, 4\}$, and $G_1\pi G_2$ is defined similarly. Note that $(G_1\sigma G_2)\sigma G_3 \neq G_1\sigma(G_2\sigma G_3)$, but $(G_1\pi G_2)\pi G_3 = G_1\pi(G_2\pi G_3)$. Given k s.p. graphs $G_1, G_2, \ldots, G_k$, we use $G_1\sigma^i G_2\sigma^i \cdots \sigma^i G_k$ to denote the s.p. graph $(((G_1\sigma^i G_2)\sigma^i G_3)\cdots \sigma^i G_k)$, and $G_1\pi^i G_2\pi^i \cdots \pi^i G_k$ to denote the s.p. graph $(((G_1\pi^i G_2)\pi^i G_3)\cdots \pi^i G_k)$. Moreover, we define $G_1\sigma G_2\sigma\cdots\sigma G_k$ as the union of $(((G_{i_1}\sigma G_{i_2})\sigma G_{i_3})\cdots \sigma G_{i_k})$, where $i_1, i_2, \ldots, i_k$ form a permutation of $1, 2, \ldots, k$. We define $G_1\pi G_2\pi\cdots\pi G_k$ similarly.

For each trail cover class of the series type, it is easy to find a corresponding dual trail cover class. Therefore, it suffices to consider a series connection of trail cover classes only. Let $TC = \{a, b, \ldots, p\}$ be the set of all trail cover classes. We define an operation σ^1 on k trail cover classes as a series connection of k trail cover classes, which maps from $TC \times \cdots \times TC$ to $P(TC)$, the power set of TC, as follows:

$x_1\sigma^1 x_2\sigma^1 \cdots \sigma^1 x_k = \{\gamma \in TC \mid$ for every s.p. graph pair (G_i, G_i') with $i = 1, 2, \ldots, k$ having a trail cover T_i in x_i, there exists an s.p. graph pair (G, G'), with $G \in G_1\sigma^1 G_2\sigma^1 \cdots \sigma^1 G_k$ and $G' \in G_1'\pi^1 G_2'\pi^1 \cdots \pi^1 G_k'$, having a trail cover T in γ such that the induced trail cover of T in G_i is T_i for all $i\}$.

The operation $x_1\sigma^1 x_2\sigma^1 \cdots \sigma^1 x_k$ can be treated as a concatenation of k trail cover classes $x_1, x_2, \ldots, x_k$ in specific series connection of s.p. graphs. For $X_1, X_2, \ldots, X_k \subseteq TC$, we define $X_1\sigma^1 X_2\sigma^1 \cdots \sigma^1 X_k$ as the union of $x_1\sigma^1 x_2\sigma^1 \cdots \sigma^1 x_k$ with $x_i \in X_i$ for every i.

We define $x_1\sigma x_2\sigma \cdots \sigma x_k$ as we define $x_1\sigma^1 x_2\sigma^1 \cdots \sigma^1 x_k$, except that G and G' in the definition are replaced by $G \in G_1\sigma G_2\sigma \cdots \sigma G_k$ and $G' \in G_1'\pi G_2'\pi \cdots \pi G_k'$. It can be treated as a series concatenation of k trail cover classes without restriction of the order of the series connection of the corresponding k s.p. graphs. It follows that $x_1\sigma^1 x_2\sigma^1 \ldots \sigma^1 x_k \subseteq x_1\sigma x_2\sigma \cdots \sigma x_k$. We define $X_1\sigma X_2\sigma \cdots \sigma X_k$ likewise. We are interested, in particular, in $x_1\sigma x_2$. Three examples are given below to demonstrate the derivation of $x_1\sigma x_2$, each also illustrated by a figure with graph pairs (G_i, G_i'). We use s_i, t_i and s_i', t_i' to denote the distinguished vertices of G_i and G_i', respectively.

EXAMPLE 2.1. We show here the derivation of $[2,2]\sigma[2,2]$, i.e., $a\sigma a$. It is known that $[2,2] = \{(t,t)/(s,s); (t,s)/(s,t)\}$. Then

$$\begin{aligned}[2,2]\sigma[2,2] = {} & \{(t,t)/(s,t); (t,s)/(s,s)\} \\ & \cup \{(t,t)/(I,I) + (s,t)/(I,I); \\ & \quad (t,s)/(I,I) + (s,s)/(I,I)\}\end{aligned}$$

$$\cup\{(t,t)/(I,I) + (s,s)/(I,I);$$
$$(t,s)/(I,I) + (s,t)/(I,I)\}$$
$$= \{[2,0],[\theta x, \phi x],[\theta x, \theta x]\}.$$

Hence $a\sigma a = \{b, k, l\}$.

Example 2.1 is illustrated in Figure 8, which contains three graph pairs. (G_1, G_1') and (G_2, G_2') possess trails $L_1 = \mathbf{a}$ and $L_2 = \mathbf{b}$, respectively, both in trail cover class a. Up to isomorphism, there is only one graph pair (G_3, G_3') with $G_3 \in G_1 \sigma G_2$ and $G_3' \in G_1' \pi G_2'$, as shown in Figure 8c. Let z be the only nondistinguished vertex in G_3. Let M be a DET network realizing a DET trail (L, L') in $(G[M], G[M'])$ that contains (G_3, G_3') as a subgraph pair. Suppose that L begins at any vertex u in $G[M]$ with $u \neq z$. Then L enters G_3 from s_3 to t_3 (or t_3 to s_3), i.e., the trail induced by L in G_3 is given by **ab**. On the other hand, L' enters G_3' either from s' to s' or from t' to t'. The trail cover induced by (L, L') in (G_3, G_3') is in class b. Suppose that L begins at z. Since z is an internal vertex in G_3, it follows that $\deg(z)$ is even and thus L terminates at z. We assume without loss of generality that L begins with **a** and terminates with **b**. We can also assume that L' begins at s'. It follows that L' first traverses G_3' by **a** and leaves at t_3' and reenters G_3' with **b**. The reentering vertex of L' in G_3' is either s_3' or t_3'. The trail cover of (L, L') induced in (G_3, G_3') is in class l if the reentering vertex is s_3' and in class k otherwise. Thus $a\sigma a = \{b, k, l\}$.

EXAMPLE 2.2. We show here the derivation of $[0,2]\sigma[0,2]$, i.e., $c\sigma c$. It is known that $[0,2] = \{(t,t)/(t,s); (s,t)/(s,s)\}$. Then

$$[0,2]\sigma[0,2] = \{(t,t)/(t,s) + (s,t)/(s,s)\} \cup \{(I,I)/(I,I)\}$$
$$= \{[\theta, \phi],[y,y]\}.$$

Hence $c\sigma c = \{d, p\}$.

Example 2.2 is illustrated in Figure 9, which contains four graph pairs. (G_1, G_1') and (G_2, G_2') possess trails $L_1 = \mathbf{ab}$ and $L_2 = \mathbf{cd}$, respectively, both in trail cover class c. Up to isomorphism, there are exactly two graph pairs (G_3, G_3') and (G_4, G_4'), with $G_i \in G_1 \sigma G_2$ and $G_i' \in G_1' \pi G_2'$ for $i = 3, 4$, as shown in Figures 9c and 9d. Let z be the only nondistinguished vertex in $G_3 (= G_4)$. Let M be a DET network realizing a DET trail (L, L') in $(G[M], G[M'])$ that contains (G_3, G_3') as a subgraph pair. Suppose that L begins at any vertex u in $G[M]$ with $u \neq z$. It follows that L enters G_3 from either s_3 or t_3. We can assume without loss of generality that L enters G_3 at s_3 with L_1. Since L_1 is in class c, L leaves G_3 at s_3 after traversing L_1 and reenters G_3 from t_3 with L_2. On the other hand, we can verify that the trail set formed by $\{(\mathbf{ab}), (\mathbf{cd})\}$ in G_3' is in class ϕ. Thus the trail cover in (G_3, G_3') induced by (L, L') is in class d. Suppose

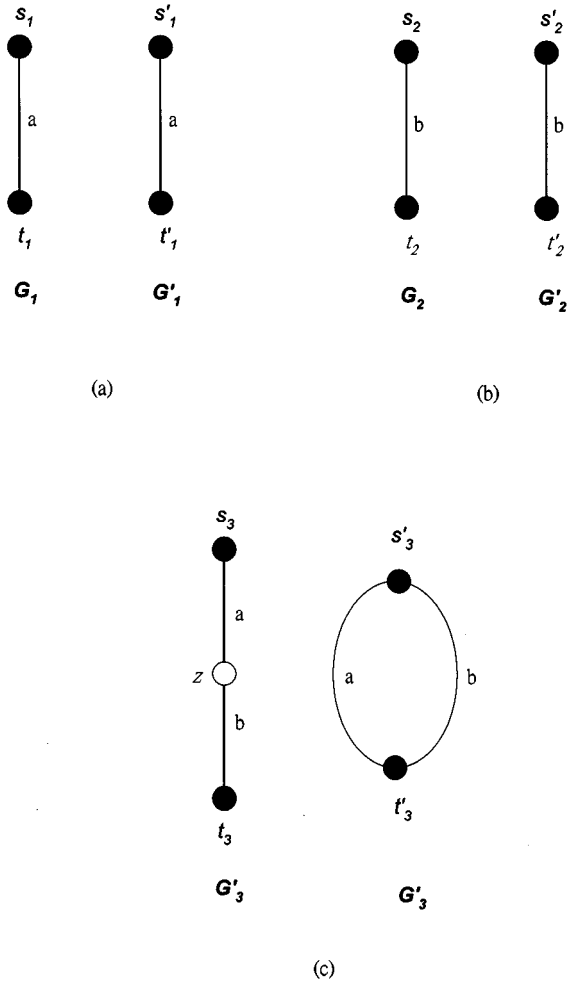


FIG. 8. Illustration of $a\sigma a = \{b, k, l\}$.

that L begins at z. We can assume without loss of generality that L begins with L_1. Since the trail cover class of L_1 is c, L_1 terminates at z and is immediately followed by L_2. Since L_2 is in c, L_2 terminates at z. Thus $L = L_1L_2$ is in class p.

Let M be a DET network realizing a DET trail (L, L') in $(G[M], G[M'])$ that contains (G_4, G'_4) as a subgraph pair. Suppose that L begins at any vertex u in $G[M]$ with $u \neq z$. As in the above case, the trail cover class in G_4 derived from L can be shown to be in class d. Suppose that L begins at

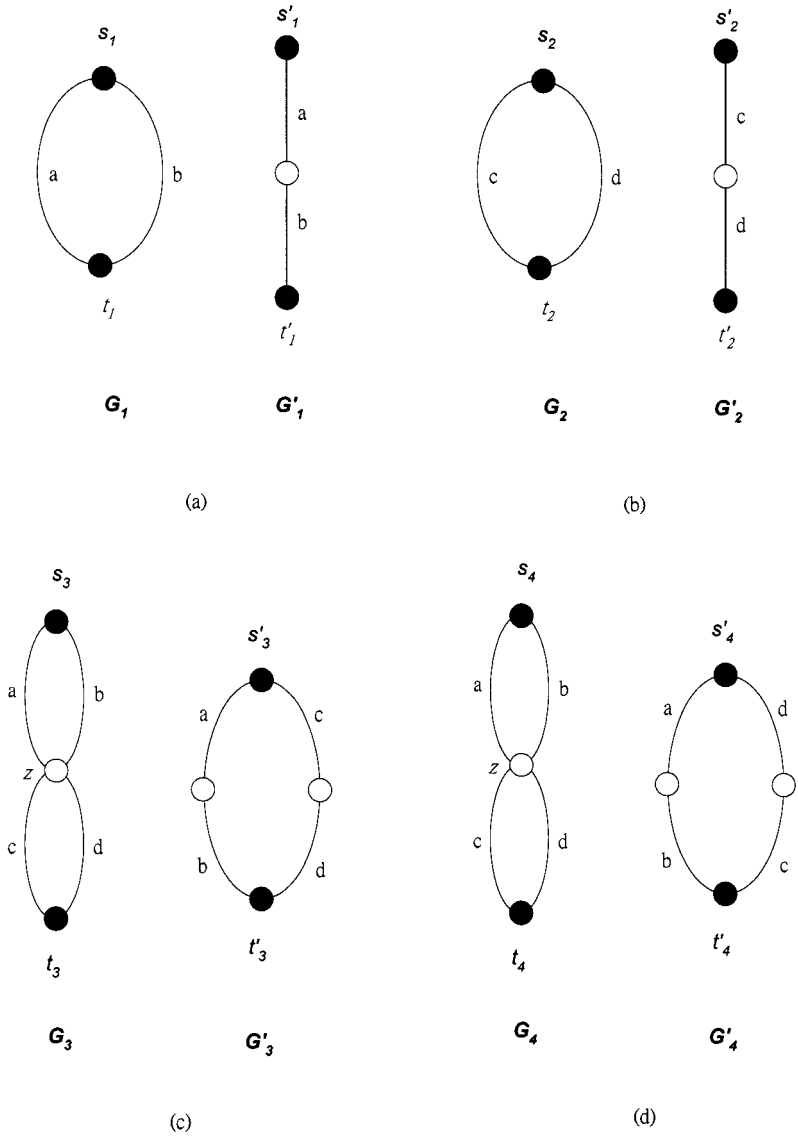


FIG. 9. Illustration of $c\sigma c = \{d, p\}$.

z. Similarly, the trail cover in G_4 derived from L is given by $L = L_1L_2$ and is in class p. Thus $c\sigma c = \{d, p\}$.

EXAMPLE 2.3. We show here the derivation of $[0, 2]\sigma[x, x]$, i.e., $c\sigma f$. It is known that $[0, 2] = \{(t, t)/(t, s); (s, t)/(s, s)\}$ and $[x, x] = \{(t, t)/(I, I);$

$(t,s)/(I,I);(s,t)/(I,I);(s,s)/(I,I)\}$. Then

$$\begin{aligned}[0,2]\sigma[x,x] = \{&(t,t)/(t,s) + (s,t)/(I,I);\\ &(t,t)/(t,s) + (s,s)/(I,I);\\ &(s,t)/(s,s) + (t,t)/(I,I);\\ &(s,t)/(s,s) + (t,s)/(I,I)\}\\ &\cup\{(I,I)/(I,I)\}\\ = \{&[(x,x) + (0,2)],[y,y]\}.\end{aligned}$$

Hence $c\sigma f = \{h, p\}$.

We illustrate Example 2.3 in Figure 10, which contains four graph pairs. (G_1, G_1') possesses a trail $L_1 = \mathbf{ab}$ in class c, and (G_2, G_2') possesses a trails $L_2 = \mathbf{cde}$ in class f. Up to isomorphism, there are four graph pairs (G, G') with $G \in G_1\sigma G_2$ and $G' \in G_1'\pi G_2'$. We here only show two graph pairs (G_i, G_i') with $G_i \in G_1\sigma G_2$ and $G_i' \in G_1'\pi G_2'$ for $i = 3, 4$. Let z_i be the common vertex of edges **e**, **c**, and **d** in G_i for $i = 3, 4$. Let M be a DET network realizing a DET trail (L, L') in $(G[M], G[M'])$ that contains (G_3, G_3') as a subgraph pair. Since $\deg(z_3) = 3$, we can assume that L begins at z_3. Thus L begins with L_2, which is followed by L_1. The trail cover in G_3 derived from L is given by $L = L_1L_2$ and is in class p. Let M be a DET network realizing a DET trail (L, L') in $(G[M], G[M'])$ that contains (G_4, G_4') as a subgraph pair. Since $\deg(z_4) = 3$, we can assume that L begins at z_4 by L_2 and leaves G_4 at t_4. Eventually L must return to G_4 at s_4 (since **e** has been traversed, t_4 is ruled out), from which it follows L_1. After traversing L_1, it will leave G_4 at s_4. Thus the trail cover class in G_4 derived from L is class h. We can use similar arguments to discuss the other two graph pairs (G, G') with $G \in G_1\sigma G_2$ and $G' \in G_1'\pi G_2'$. Consequently, we obtain $c\sigma f = \{h, p\}$.

Using arguments similar to that employed in the above examples, we construct Table 1 to illustrate the results of σ on $TC \times TC$. If a series connectin of two trail cover classes does not yield a trail cover in TC, we write $\alpha\sigma\beta = \varnothing$, which is indicated by a blank entry in Table 1. Obviously, $\tau_G(N, \mathscr{L})$ is in trail cover class a if L is a DET and N is of type L. Note that Table 1 is closed under series connection, parallel connection, and taking the duals generated by class a. There are exactly 16 different trail cover classes needed to form DET.

A network N is *possible* if there exists a trail cover class in TC for some $(G[N], G[N'])$. We use Q to denote the set of all possible networks. For any network N in Q, $Possible(N) = \{(G[N], G[N']) \mid$ there exists a trail cover for $(G[N], G[N'])$, which is in $Q\}$, and $Class(N) = \{z \mid z$ is a trail

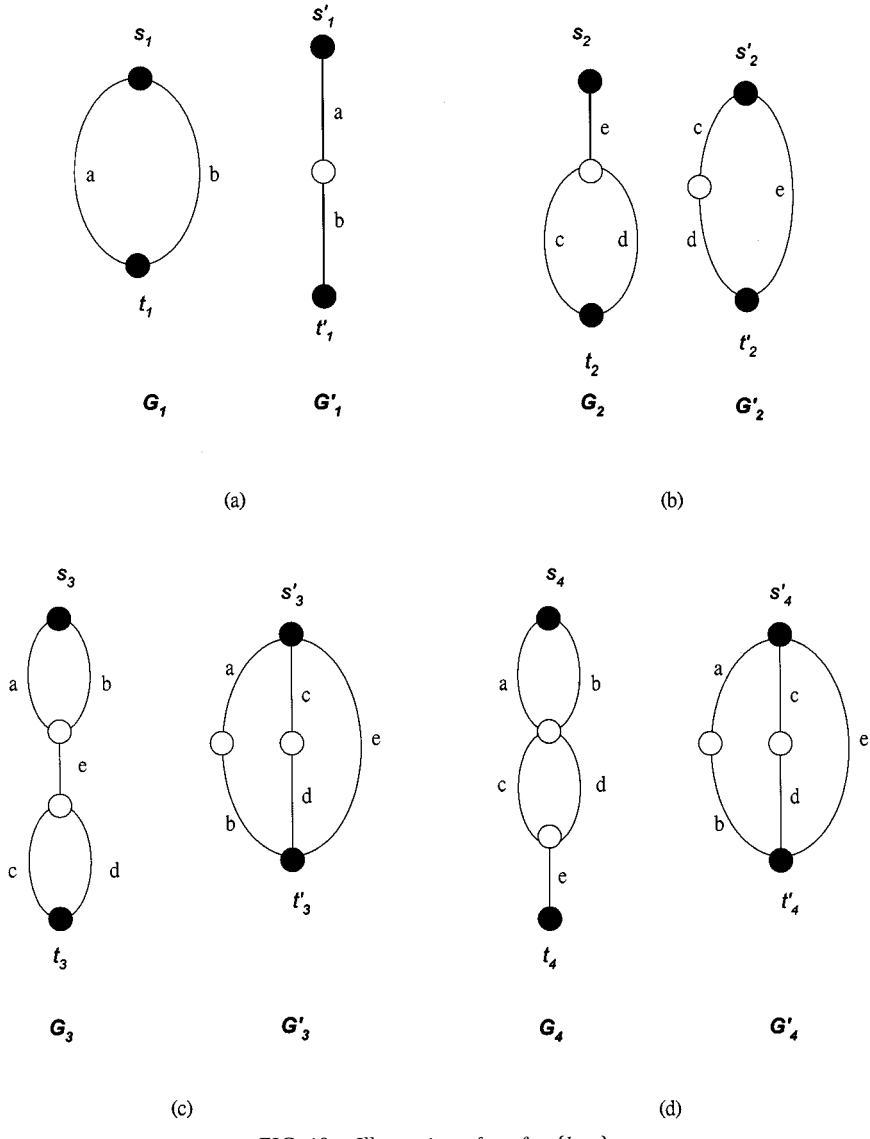


FIG. 10. Illustration of $c\sigma f = \{h, p\}$.

cover class for some $(G[N], G[N'])$ in $Possible(N)\}$. For example, $Class(N) = \{a\}$ for any network N of type L. It follows from Example 2.1 that $Class(N) = \{b, k, l\}$ for any network N that is a series connection of two networks of type L. Hence we have $Class(N) = \{c, j, l\}$ for any network N that is a parallel connection of two networks of type L.

TABLE 1
The function σ that maps $TC \times TC$ to $P(TC)$

set / σ	T_A		T_C	T_D	T_E	T_F										
	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*	*n*	*o*	*p*
T_A *a*	*bkl*	*akl*	*fh*	*h*	*g*	*fkl*	*il*	*hkl*	*gk*		*l*	*k*				
T_A *b*	*akl*	*bkl*	*fh*	*h*	*i*	*fkl*	*gk*	*hkl*	*il*		*k*	*l*				
T_C *c*	*fh*	*fh*	*dp*		*cjn*	*hp*	*fjn*		*fjn*	*np*			*j*		*j*	
T_D *d*	*h*	*h*			*dn*		*hn*		*hn*				*n*		*n*	
T_E *e*	*g*	*i*	*cjn*	*dn*	*eo*	*j*	*gm*	*hn*	*io*	*j*			*m*	*n*	*o*	
T_F *f*	*fkl*	*fkl*	*hp*		*j*	*klp*	*j*		*j*							
T_F *g*	*ij*	*gk*	*fjn*	*hn*	*gm*	*j*	*lo*	*kln*	*km*							
T_F *h*	*hkl*	*hkl*			*hn*		*kln*		*kln*							
T_F *i*	*gk*	*il*	*fjn*	*hn*	*io*	*j*	*km*	*kln*	*lo*							
T_F *j*			*np*		*j*											
T_F *k*	*l*	*k*														
T_F *l*	*k*	*l*														
T_F *m*			*j*	*n*	*m*											
T_F *n*					*n*											
T_F *o*			*j*	*n*	*o*											
	p															

EXAMPLE 2.4. Let N be the network in Figure 1, N_1 be the child subnetwork of N defined on {**e**}, N_2 be the child subnetwork defined on {**a**, **b**}, and N_3 be the child subnetwork of N defined on {**c**, **d**}. Then $Class(N_1) = \{a\}$, $Class(N_2) = Class(N_3) = \{c, j, l\}$. It follows from the discussion in Examples 2.1, 2.2, and 2.3 that $Class(N_1)\sigma^1 Class(N_2)\sigma^1 Class(N_3) = \{h\}$ and $Class(N_1)\sigma Class(N_2)\sigma Class(N_3) = \{h, p\}$. Therefore, the network in Figure 1 has a DET trail in class p.

Let N be a possible network of type S with child subnetworks $N_1, N_2, \ldots, N_k$. It can be observed from the above example that $Class(N) = Class(N_1)\sigma Class(N_2)\sigma \cdots \sigma Class(N_k)$. Thus we need methods to compute $Class(N)$ from its child subnetworks.

3. PROPERTIES OF DET NETWORKS

Using Table 1, we can derive the generation of classes a, b, c, d, and e.

LEMMA 3.1. *The classes a, b, c, d and e of type S are constructed as follows*:

(i) $[2, 2]_S \leftarrow \{m[2, 2]_P\}\sigma\{n[2, 0]_P\}$, *where m is odd and* $m + n \geq 2$.

(ii) $[2, 0]_S \leftarrow \{m[2, 2]_P\}\sigma\{n[2, 0]_P\}$, *where m is even and* $m + n \geq 2$.

(iii) $[0, 2]_S \leftarrow [0, 2]_P\,\sigma\{m[\phi, \theta]_P\}$, *where* $m \geq 1$.

(iv) $[\theta, \phi]_S \leftarrow [0, 2]_P\,\sigma[0, 2]_P\,\sigma\{m[\phi, \theta]_P\}$.
$[\theta, \phi]_S \leftarrow \{m[\phi, \theta]_P\}\sigma[\theta, \phi]_P$, *where* $m \geq 1$.

(v) $[\phi, \theta]_S \leftarrow \{m[\phi, \theta]_P\}$, *where* $m \geq 2$.

In these rules, by "$z[-, -]_P$*" we mean a series connection of* $z[-, -]_P$*'s where z is a nonnegative integer. The subscript S* (P) *is used to indicate a trail cover for some networks of type S* (P).

Proof. Every network is recursively constructed from edges and subnetworks that correspond to leaves and subtrees, respectively. Each edge has a trail cover $[2, 2]$ only of type P or S, depending on the type of its parent. From Table 1, we know that $a = a\sigma b$, $b = a\sigma a$ or $b\sigma b$, $c = c\sigma e$, $d = d\sigma e$ or $c\sigma c$, and $e = e\sigma e$. Repeatedly applying these relations, we obtain the rules as stated in the lemma. ∎

Examining Lemma 3.1 and its proof, we find that the classes a, b, c, d, and e are generated by concatenation of these five classes only, without involving the remaining 11 trail cover classes. Hence we call classes a, b, c, d, and e *primary trail cover classes*, and any trail cover in classes a, b, c, d, and e a *primary trail cover*.

LEMMA 3.2. *The classes a, b, c, d, and e are mutually exclusive. In other words, let network N have a trail cover in* $\alpha \in \{a, b, c, d, e\}$. *Then N cannot have a trail cover* β *such that* $\beta \in \{a, b, c, d, e\}$ *and* $\alpha \neq \beta$.

Proof. According to Lemma 3.1, the generation of a, b, c, d, and e are mutually exclusive. ∎

To simplify our exposition, we divide the 15 trail cover classes $a, b, \ldots, o$, excluding p, of type S and of type P into five groups:

$$T_A = \{[2,2]_P, [2,0]_P, [2,2]_S, [0,2]_S\}$$

$$T_C = \{[0,2]_P, [2,0]_S\}$$

$$T_D = \{[\theta, \phi]_P, [\phi, \theta]_S\}$$

$$T_E = \{[\phi, \theta]_P, [\theta, \phi]_S\}$$

$$\begin{aligned} T_F = \{&[x,x]_P, [(x,x)+(2,2)]_P, [(x,x)+(0,2)]_P, [(x,x)+(2,0)]_P,\\ &[x,x]_S, [(x,x)+(2,2)]_S, [(x,x)+(2,0)]_S, [(x,x)+(0,2)]_S,\\ &[\phi x, \theta x]_P, [\theta x, \phi x]_P, [\theta x, \theta x]_P, [(\theta x, \theta x)+(2,2)]_P,\\ &[\theta x, \phi x]_S, [\phi x, \theta x]_S, [\theta x, \theta x]_S, [(\theta x, \theta x)+(2,2)]_S,\\ &[(\phi x, \theta x)+(0,2)]_P, [(\theta x, \phi x)+(2,0)]_P,\\ &[(\theta x, \phi x)+(2,0)]_S, [(\phi x, \theta x)+(0,2)]_S\}. \end{aligned}$$

It follows from Lemma 3.2 that trail covers in T_A, T_C, T_D, and T_E are primary trail cover classes and are mutually exclusive. As shown in Figure 6, the trail cover types of $\tau_G(N_i, \{L\})$ for $i = 1, 10$ are in T_C. The trail cover types of $\tau_G(N_i, \{L\})$ for $i = 2, 3, 9$ are in T_E. The trail cover types of $\tau_G(N_i, \{L\})$ for $i = 4, 5, 6, 7$ are in T_A. The trail cover type of $\tau_G(N_8, \{L\})$ is in T_F. In general, let $(G[M], G[M'])$ be any graph representation that possesses a DET trail L for some network M. Let N be any type S subnetwork of M. In the following, we will show that those child s.p. subgraphs of $G_M[N]$ with their derived trail cover classes (from L) in T_A induce at most two connected components. There are at most two child s.p. subgraphs of $G_M[N]$ with their derived trail cover classes in T_C. There is at most one child s.p. subgraph of $G_M[N]$ with its derived trail cover class in T_D. Moreover, if there is a child s.p. subgraph of $G_M[N]$ with its derived trail cover class in T_C, then there is no child s.p. subgraph of $G_M[N]$ with its derived trail cover class in T_E. Those child s.p. subgraphs of $G_M[N]$ with their derived trail cover classes in T_E induce at most two connected components. There are at most two child s.p. subgraphs of $G_M[N]$ with their derived trail cover classes in T_F.

Let M be a DET network that possesses a DET (L, L') in $(G[M], G[M'])$. Let N be a subnetwork of M. Since M is DET, it suffices to consider properties of trail cover classes of M and of N only, without

considering M' and N'. In the following sections, we use a DET L in $G[M]$ to represent a DET (L, L') in $(G[M], G[M'])$ without ambiguity. We would like to study the properties of subnetworks of a DET network.

LEMMA 3.3. *Let (G, G') realize a DET L for network M, and let N be a subnetwork of M.*

(i) *Let N_1 be a child subnetwork of N such that $L_G[N_1]$ is a trail cover in T_F. Then L must begin or end at an edge in N_1.*

(ii) *N has at most two child subnetworks N_i such that each $L_G[N_i]$ is a trail cover in T_F.*

Proof. Each nondistinguished trail contains at least one internal vertex as a terminal, and therefore one of its termini cannot be concatenated with a trail cover of another subnetwork. Trail covers in T_F contain at least one nondistinguished trail. Consequently, these nondistinguished trails must be the beginning section or the ending section of L. Therefore, statement (i) follows. If N contains more than two child subnetworks N_i such that each $L_G[N_i]$ is a trail cover in T_F, N contains more than two nondistinguished trails. Since trail covers in TC contain at most two nondistinguished trails, N cannot have any trail cover in TC. Furthermore, M cannot be DET, which contradicts the assumption. Hence the lemma follows. ▌

LEMMA 3.4. *Let M be a DET network, $G[M]$ realize a DET L, and N be a type S subnetwork of M. Then the following statements hold*:

(i) *N can contain at most two child subnetworks C_i such that each $L_M[C_i]$ is a trail cover in T_C.*

(ii) *N can contain at most one child subnetwork D such that $L_M[D]$ is a trail cover in T_D.*

(iii) *N cannot contain both child subnetworks C_i and D, where C_i and D are as defined above.*

Proof. To prove statement (i), we assume without loss of generality that there are exactly three child subnetworks C_1, C_2, and C_3 of N such that each $L_G[C_i]$ is a trail cover in T_C. Then $G_M[N]$ can be written as

$$G_M[N] = G_M[N_1]\sigma^1 G_M[C_1]\sigma^1 G_M[N_2]\sigma^1 G_M[C_2]$$
$$\sigma^1 G_M[N_3]\sigma^1 G_M[C_3]\sigma^1 G_M[N_4],$$

where N_i can be vacuous for $i = 1, 2, 3, 4$. Since $c\sigma^1 c\sigma^1 c \subseteq c\sigma c\sigma c = \varnothing$, it follows that $G_M[C_1]$, $G_M[C_2]$, and $G_M[C_3]$ cannot all be placed consecutively in $G[M]$, i.e., N_2 and N_3 cannot both be vacuous. Next, we consider the case where only two $G_M[C_i]$ are placed consecutively in $G_M[N]$. Suppose that $G_M[C_1]$ and $G_M[C_2]$ are connected, i.e., N_2 is vacuous. The

possible trail covers of $G_M[N]$ are contained in the set $\tau_1 = (TC - \{c\})\sigma^1\{c\}\sigma^1\{c\}\sigma^1(TC - \{c\})\sigma^1\{c\}\sigma^1(TC - \{c\})$. It follows from Table 1 that

$$\begin{aligned}\tau_1 &\subseteq \big(\big(\big(\big(\big((TC - \{c\})\,\sigma\{c\}\big)\,\sigma\{c\}\big)\sigma(TC - \{c\})\big)\sigma\{c\}\big)\sigma(TC - \{c\})\big)\\ &= \big(\big(\big(\big(\{c, f, h, j, n, p\}\sigma\{c\}\big)\,\sigma(TC - \{c\})\big)\sigma\{c\}\big)\sigma(TC - \{c\})\big)\\ &= \big(\big(\big(\{d, h, n, p\}\sigma(TC - \{c\})\big)\sigma\{c\}\big)\sigma(TC - \{c\})\big)\\ &= \big(\{d, h, k, l, n\}\sigma\{c\}\sigma(TC - \{c\})\big) = \varnothing.\end{aligned}$$

Thus there is no trail cover in $G_M[N]$ for $G[M]$ to form a DET, i.e., N does not have a trail cover in TC. Finally, we consider the case where N_2 and N_3 are not vacuous. In this case, the possible trail covers of $G_M[N]$ are contained in the set $\tau_2 = (TC - \{c\})\sigma^1\{c\}\sigma^1(TC - \{c\})\sigma^1\{c\}\sigma^1(TC - \{c\})\sigma^1\{c\}\sigma^1(TC - \{c\})$. As in the first case, we can show that the set τ_2 is empty. Therefore statement (i) follows.

Statements (ii) and (iii) can be proved by arguments similar to those given for statement (i). ▮

LEMMA 3.5. *Let M be a DET network, $G[M]$ realize a DET L, N be a type S subnetwork of M, and $A_1, \ldots, A_n$ be the only child subnetworks of N such that each $L_G[A_i]$ is in T_A. Then in $G_M[N]$, $A_1, \ldots, A_n$ are represented by at most two connected subgraphs.*

Proof. Suppose that in $G_M[N]$, the subnetworks $A_1, A_2, \ldots, A_n$ are represented by three connected components, say, $G_M[K_1]$, $G_M[K_2]$, and $G_M[K_3]$. It follows that $G_M[N]$ can be written as

$$\begin{aligned}G_M[N] = G_M[N_1]\sigma^1 G_M[K_1]\sigma^1 G_M[N_2]\sigma^1 G_M[K_2]\\ \sigma^1 G_M[N_3]\sigma^1 G_M[K_3]\sigma^1 G_M[N_4],\end{aligned}$$

where N_1 and N_4 can be vacuous. As in the deduction in Example 2.1, we have $T_A\sigma^1 T_A = T_A$. It follows that each $L_G[K_i]$ is in T_A for every i. By Table 1, T_A can be obtained only by $T_A\sigma T_A$. It follows that the induced trail cover in each $G_M[N_i]$ is contained in $Y = TC - \{a, b\}$. Moreover, N_2 and N_3 cannot be vacuous, since otherwise the A_i's can be represented by less than three connected components. Therefore the induced trail cover in $L_G[N]$ is contained in $Y\sigma^1\{a, b\}\sigma^1 Y\sigma^1\{a, b\}\sigma^1 Y\sigma^1\{a, b\}\sigma^1 Y$. It is obvious that $Y\sigma^1\{a, b\}\sigma^1 Y \subseteq Y\sigma\{a, b\}\sigma Y$ and $Y\sigma^1 Y\sigma^1\{a, b\} \subseteq Y\sigma^1\{a, b\} \subseteq Y\sigma\{a, b\}$. The trail covers in $Y\sigma\{a, b\}$ are concatenated with trail covers in other subnetworks by means of a trail cover in $\{a, b\}$, while the trail covers in $Y\sigma^1\{a, b\}\sigma^1 Y$ are concatenated by means of both trail covers in Y. Since a and b are primary trail cover classes and $Y \cap \{a, b\} = \varnothing$, it follows that

$(Y\sigma^1\{a, b\}\sigma^1 Y) \cap (Y\sigma\{a, b\}) = \emptyset$. Therefore, $Y\sigma^1\{a, b\}\sigma^1 Y \subseteq (Y\sigma\{a, b\}\sigma Y) - (Y\sigma\{a, b\}) = \{j, m, n, o, p\} = W$. Moreover, $\{a, b\}\sigma^1 W\sigma^1\{a, b\} \subseteq (\{a, b\}\sigma W)\sigma\{a, b\} = \emptyset$, i.e., $L_G[N] = \emptyset$. Therefore, there exists no trail cover in $G_M[N]$ for $G[M]$ to form a DET, i.e., N has no trail cover in TC. This leads to a contradiction, and hence the lemma follows. ▮

LEMMA 3.6. *Let M be a DET network, $G[M]$ realize a DET L, N be a type S subnetwork of M, and $E_1, \ldots, E_n$ be the only child subnetworks of N such that each $L_G[E_i]$ is in T_E. Then in $G_M[N]$, $E_1, \ldots, E_n$ are represented by at most two connected subgraphs.*

Proof. Suppose that in $G_M[N]$, the subnetworks $E_1, E_2, \ldots, E_n$ are represented by three connected components, say, $G_M[K_1]$, $G_M[K_2]$, and $G_M[K_3]$. It follows that $G_M[N]$ can be written as

$$G_M[N] = G_M[N_1]\sigma^1 G_M[K_1]\sigma^1 G_M[N_2]\sigma^1 G_M[K_2]$$
$$\sigma^1 G_M[N_3]\sigma^1 G_M[K_3]\sigma^1 G_M[N_4],$$

where N_1 and N_4 can be vacuous. Similar to the deduction in Example 2.1, we have $T_E\sigma^1 T_E = T_E$. It follows that each $L_G[K_i]$ is in T_E for every i. By Table 1, T_E can be obtained only by $T_E\sigma T_E$. It follows that the induced trail covers in each $G_M[N_i]$ are contained in $Y = TC - \{e\}$. Moreover, N_2 and N_3 cannot be vacuous, since otherwise the E_i's can be represented by less than three connected components. Therefore the induced trail cover in $L_G[N]$ is contained in $Y\sigma^1\{e\}\sigma^1 Y\sigma^1\{e\}\sigma^1 Y\sigma^1\{e\}\sigma^1 Y$. It is obvious that $Y\sigma^1\{e\}\sigma^1 Y \subseteq Y\sigma\{e\}\sigma Y$ and $Y\sigma^1 Y\sigma^1\{e\} \subseteq Y\sigma^1\{e\} \subseteq Y\sigma\{e\}$. The trail covers in $Y\sigma\{e\}$ are concatenated with trail covers in other subnetworks by means of a trail cover in $\{e\}$, while the trail covers in $Y\sigma^1\{e\}\sigma^1 Y$ are concatenated by means of both trail covers in Y. Since $Y \cap \{e\} = \emptyset$ and e is a primary trail cover class, it follows that $(Y\sigma^1\{e\}\sigma^1 Y) \cap (Y\sigma\{e\}) = \emptyset$. Therefore, $Y\sigma^1\{e\}\sigma^1 Y \subseteq (Y\sigma\{e\}\sigma Y) - (Y\sigma\{e\}) = \{f, k, l, p\} = W$. Similarly, we have $\{e\}\sigma^1 W\sigma^1\{e\} \subseteq (\{e\}\sigma W)\sigma\{e\} = \emptyset$. Therefore, there exists no trail cover in $G_M[N]$ for $G[M]$ to form a DET, i.e., N has no trail cover in TC. This leads to a contradiction, and hence the lemma follows. ▮

LEMMA 3.7. *Let M be a DET network that realizes a DET L in $G[M]$. Then the following statements hold*:

(i) *All child subnetworks A_i of M such that each $L_G[A_i]$ is in T_A are represented by only one connected component in $G[M]$.*

(ii) *M does not contain a child subnetwork N such that $L_G[N] \in T_D$.*

Proof. To prove statement (i), suppose that all of the A_i's are represented by two connected components in $G[M]$. We use arguments similar to those in the proof of Lemma 3.5 and obtain the result that $G[M]$ can be

written as follows:

$$G[M] = G_M[N_1]\sigma^1 G_M[K_1]\sigma^1 G_M[N_2]\sigma^1 G_M[K_2]\sigma^1 G_M[N_3],$$

where N_1 and N_3 can be vacuous, N_2 cannot be vacuous, $L_G[K_i]$ is in T_A for $i = 1, 2$, and trail covers in each $G_M[N_i]$ are contained in $Y = TC - \{a, b\}$. It follows that L is contained in $Y\sigma^1\{a, b\}\sigma^1 Y\sigma^1\{a, b\}\sigma^1 Y$. It follows from repeated applications of Table 1 that $\{a, b\}\sigma^1\{c, d, e\}\sigma^1\{a, b\} \subseteq (\{a, b\}\sigma\{c, d, e\}\sigma\{a, b\}) - (\{a, b\}\sigma\{c, d, e\}) = \{k, l\}$, $\{a, b\}\sigma^1(Y - \{c, d, e\})\sigma^1\{a, b\} \subseteq (\{a, b\}\sigma(Y - \{c, d, e\})\sigma\{a, b\}) - (\{a, b\}\sigma(Y - \{c, d, e\})) = \emptyset$. Therefore, we have $\{a, b\}\sigma^1 Y\sigma^1\{a, b\}\sigma^1 Y \subseteq \{k, l\}\sigma Y = \emptyset$. This contradicts the existence of L, and hence statement (i) follows.

To prove statement (ii), we suppose that M has a child subnetwork N such that $L_G[N] \in T_D = \{d\}$. It follows from Table 1 that we have $d\sigma TC = \{d, h, n\}$, $d\sigma TC\sigma TC \subseteq \{d, h, n\}\sigma TC = \{d, h, k, l, n\}$, and $d\sigma TC\sigma TC\sigma TC \subseteq \{d, h, k, l, n\}\sigma TC = \{d, h, k, l, n\}$. Therefore, the possible trail covers in M are contained in d, h, k, l, n and, as a result, M is not DET. This leads to a contradiction. Hence the lemma follows. ∎

Remark 3.1. Statement (i) of Lemmas 3.4 and 3.6 also holds for $N = M$. In other words, let M be a DET network that realizes a DET L in $G[M]$. Then M contains at most two child subnetworks C_i such that each $L_G[C_i]$ is in T_C. All of the child subnetworks E_i of M such that each $L_G[E_i]$ is in T_E are represented by at most two connected components in $G[M]$.

4. RULES FOR TRAIL COVER CLASS COMPOSITION AND REFINEMENT

In this section we propose an algorithm that generates the rules for the composition of each trail cover class. These rules are used to find all trail cover classes of a network in all graph representations. For the rules for trail cover composition, it suffices to consider series connections of trail cover classes. For a network of the parallel type, we take the duals of its child subnetworks and apply the rules for series type. Then we again take the duals of the resulting trail cover classes to obtain the trail cover classes for the network.

On the basis of the analysis in Section 3, we use an array A of size 8 to represent all trail cover classes. Elements of A are labeled by a_1, a_2, c_1, d_1, e_1, e_2, f_1, and f_2. According to Lemma 3.5, subnetworks with trail covers in T_A are represented by at most two connected components in $G[M]$, or else they cannot form a trail cover in TC. Therefore, we use a_1

and a_2 to represent all possible representations of trail covers in T_A; each element corresponds to a connected component. a_1 is used to indicate three possibilities regarding trail covers in T_A, i.e., none in T_A, a trail cover in class a, and a trail cover in class b. Lemma 3.1 states that class a consists of an odd number of $[2, 2]_P$ and some $[2, 0]_P$, and class b consists of an even number of $[2, 2]_P$ and some $[2, 0]_P$. We therefore define $a_1 = 0$ to mean no trail cover in T_A, $a_1 = 1$ for an odd number of $[2, 2]_P$, and $a_1 = 2$ for an even number of $[2, 2]_P$. Similarly, we let $a_2 = 0$ mean no trail cover in T_A, $a_2 = 1$ an odd number of $[2, 2]_P$, and $a_2 = 2$ an even number of $[2, 2]_P$ for the second connected component. When subnetworks with trail covers in T_A are represented by only one connected component in $G[M]$, we define $a_2 = 0$.

By Lemma 3.4, a network can have at most two child subnetworks with trail covers in T_C. We write $c_1 = 0$ for no trail cover in T_C, $c_1 = 1$ for one in T_C, and $c_1 = 2$ for two.

By Lemma 3.4, we define $d_1 = 0$ for no trail cover in T_D, $d_1 = 1$ for one in T_D.

As in the case of trail covers in T_A, each of e_1 and e_2 corresponds to a connected component. For the first component, we use $e_1 = 0$ to indicate no trail cover in T_E and $e_1 = 1$ to indicate at least one in T_E. We define e_2 similarly.

According to Lemma 3.3, a network contains at most two child subnetworks with trail covers in T_F. Therefore, we use f_1 and f_2 to represent the trail cover classes in T_F, excluding p, since it cannot be concatenated with any other trail cover class. Each of f_1 and f_2 takes one of 11 possible values: 0 for no trail cover in T_F, and $1, 2, \ldots, 10$ for the 10 trail cover classes in T_F (see Table 1).

We consider all of the combinations of trail cover composition in array A. For each combination, take all permutations to obtain a series connection of these trail cover classes with different orders of concatenation, since each permutation corresponds to a fixed order of concatenation of corresponding graph representations. Since there are only eight elements and each takes of a small number of different values, there are a finite number of permutations. Using Table 1, we find all possible trail cover classes for each permutation and store the results. Finally, we sort these results to obtain the composition of each trail cover class.

We summarize the above discussion in the following procedure:

Step 1. Set up values for A as discussed above. For each combination of these eight arrays, get all permutations and find the resulting trail cover classes according to Table 1.

Step 2. Sort these results to obtain the composition for each trail cover class.

It can be easily verified that there are at most $(8!/(2!2!2!))(3^2)(3^1)(2^1)(2^2)(11^2)$ permutations in the above procedure. Therefore, this procedure can be carried out in constant time. Since some permutations cannot yield a trail cover in classes a to p, the number of rules can be greatly reduced. In Table 2 we list a total of 156 rules generated from this procedure. The trail cover classes listed in each rule are arranged in order from top to bottom in the graph representation, so it is convenient to check Table 2 to determine the appropriate representation for each trail cover class. In Table 2, by $z[-,-]_P$ we mean series connection of $z[-,-]_P$'s, where z is a nonnegative integer.

We note from Table 2 that series connectin of some specific trail cover classes results in trail covers in more than one class, one in group $T_A \cup T_C \cup T_D \cup T_E$, and the others in T_F. For example, suppose that N_1 and N_2 have trail covers in class a. According to Table 1, trail covers for $N_1 \sigma N_2$ can be in classes b, k, l where $b \in T_A$ and $k, l \in T_F$. We want to find trail covers of $N_1 \sigma N_2 \sigma N_3$ for an arbitrary network N_3. We distinguish two cases for trail covers of $N_1 \sigma N_2$. In the first case, let $\{k, l\}$ be the resulting trail covers in $N_1 \sigma N_2$. By Table 1, N_3 must be a network with a trail cover in T_A, say a, otherwise $N_1 \sigma N_2 \sigma N_3$ cannot have any trail cover in TC. Furthermore, $N_1 \sigma N_2 \sigma N_3$ has trail covers in $\{k, l\}$. Consider the second case, where b is the resulting trail cover in $N_1 \sigma N_2$. Let N_3 be a network with a trail cover in class a as in the first case. It follows from Table 1 that $N_1 \sigma N_2 \sigma N_3$ has trail covers in $\{a, k, l\}$. Comparing the resulting trail covers of $N_1 \sigma N_2 \sigma N_3$ in both cases, we find that $\{k, l\}$ in the first case is contained in $\{a, k, l\}$ of the second case. Therefore, we can write $a \sigma a = b$ instead of $a \sigma a = \{b, k, l\}$ when considering the composition of DET or DEC networks. This fact can also be observed from Example 2.1.

On the other hand, we will show that the trail covers in b resulting from $a \sigma a$ can be obtained from the trail covers in $\{k, l\}$ and vice versa. Let M be a DET network that realizes a DET (L, L') in (G, G'), and let $N = N_1 \sigma N_2$ be a partial subnetwork of M, where $L_G[N_i]$ as in a for $i = 1, 2$. Suppose that $L_G[N]$ is in k. Let L_i denote the trail in N_i induced by L. We assume without loss of generality that L_1 is the beginning section of L, L_2 is the ending section of L, and L can be written as $L = L_1 Q L_2$. Obviously, L begins at the vertex that serially connects $G_M[N_1]$ with $G_M[N_2]$ and terminates at the same vertex. Furthermore, L' begins at one of the distinguished vertices, which connects $G_{M'}[N_1']$ and $G_{M'}[N_2']$ in parallel and terminates at the same vertex. It follows that (G, G') has another DET trail, namely, $L^* = Q L_2 L_1$. Moreover, $L_G^*[N] = L_2 L_1$ is in b. We can provide a similar proof for the case where $L_G[N]$ is in l. Now suppose that $L_G[N]$ is in b. By reversing the above operation, we can derive the trail covers in $\{k, l\}$ from the trail cover in b that results from $a \sigma a$.

TABLE 2
Rules for trail cover class composition and refinement

a $[2,2]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\}$, u *is odd and* $u+v \geq 2$.

b $[2,0]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\}$, u *is even and* $u+v \geq 2$.

c $[0,2]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\}$, $u \geq 1$.

d $[\theta,\phi]_S = [0,2]_P \sigma [0,2]_P \sigma \{u[\phi,\theta]_P\}$.
$[\theta,\phi]_S = [\theta,\phi]_P \sigma \{u[\phi,\theta]_P\}$, $u \geq 1$.

e $[\phi,\theta]_S = \{u[\phi,\theta]_P\}$, $u \geq 2$.

f $[x,x]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma \{v[2,2]_P$ *or* $w[2,0]_P\}$, $v,w \geq 1$.
$[x,x]_S = [x,x]_P \sigma \{u[2,2]_P$ *or* $v[2,0]_P\}$, $u,v \geq 1$.
$[x,x]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{v[2,2]_P\} \sigma \{w[2,0]_P\}$.
$[x,x]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{v[2,2]_P\} \sigma \{w[2,0]_P\}$.

g $[(x,x)+(2,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\}$, u *is odd*, $w \geq 1$.
$[(x,x)+(2,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}$, u *is odd.*
$[(x,x)+(2,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}$, u *is even.*

h $[(x,x)+(0,2)]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\}$, $u,v \geq 1$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [\theta,\phi]_P \sigma \{w[\phi,\theta]_P\}$, $u,v \geq 1$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [0,2]_P \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\}$, $u,v \geq 1$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(0,2)]_P \sigma \{w[\phi,\theta]_P\}$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [x,x]_P \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\}$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma [0,2]_P \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\}$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma [0,2]_P \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\}$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma [\theta,\phi]_P \sigma \{w[\phi,\theta]_P\}$.
$[(x,x)+(0,2)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma [\theta,\phi]_P \sigma \{w[\phi,\theta]_P\}$.

i $[(x,x)+(2,0)]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma \{w[\phi,\theta]_P\}$, u *is even*, $u,v,w \geq 1$.
$[(x,x)+(2,0)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}$, u *is odd.*
$[(x,x)+(2,0)]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}$, u *is even.*

j $[\phi x,\theta x]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\}$, $w \geq 1$.
$\surd$ $[\phi x,\theta x]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}$.
$\surd$ $[\phi x,\theta x]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}$.
$[\phi x,\theta x]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [(\theta x,\theta x)+(2,2)]$.
$[\phi x,\theta x]_S = [0,2]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}$.
$[\phi x,\theta x]_S = [0,2]_P \sigma [(x,x)+(2,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}$.
$[\phi x,\theta x]_S = [0,2]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}$.
$[\phi x,\theta x]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [(\theta x,\phi x)+(2,0)]_P$.
$[\phi x,\theta x]_S = [x,x]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\}$, $w \geq 1$.
$[\phi x,\theta x]_S = [x,x]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}$.
$[\phi x,\theta x]_S = [x,x]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}$.
$[\phi x,\theta x]_S = \{u[\phi,\theta]_P\} \sigma [\phi x,\theta x]_P$, $u \geq 1$.

k $[\theta x,\phi x]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\}$, $u+v \geq 2$.
* $[\theta x,\phi x]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\}$, $u+v \geq 2$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [x,x]_P$, $u,v \geq 1$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\} \sigma [(x,x)+(2,2)]_P$, $u,v \geq 1$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\} \sigma [(x,x)+(2,0)]_P$, $u,v \geq 1$.
* $[\theta x,\phi x]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [\theta,\phi]_P \sigma \{w[\phi,\theta]_P\}$, $u+v \geq 2$.
* $[\theta x,\phi x]_S = \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [0,2]_P \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\}$, $u+v \geq 2$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma \{w[\phi,\theta]_P\} \sigma [(x,x)+(0,2)]_P$, $u,v \geq 1$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma \{w[\phi,\theta]_P\} \sigma [0,2]_P \sigma [x,x]_P$, $u,v \geq 1$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [0,2]_P \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\} \sigma [(x,x)+(2,2)]_P$, $u,v \geq 1$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [0,2]_P \sigma [0,2]_P \sigma \{w[\phi,\theta]_P\} \sigma [(x,x)+(2,0)]_P$, $u,v \geq 1$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [\theta,\phi]_P \sigma \{w[\phi,\theta]_P\} \sigma [(x,x)+(2,2)]_P$, $u,v \geq 1$.
$\surd$ $[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\} \sigma [\theta,\phi]_P \sigma \{w[\phi,\theta]_P\} \sigma [(x,x)+(2,0)]_P$, $u,v \geq 1$.

TABLE 2—*Continued*

√	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,2)]_P$, *u is even,* $u+v\geq 2$.
√	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$, *u is odd,* $u+v\geq 2$.
*	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}$, *u is odd,* $u+v\geq 2$, $w\geq 1$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [\theta x,\theta x]_P$, *u is odd.*
	$[\theta x,\phi x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ [\theta x,\phi x]_P$, $u,v\geq 1$, *u is even.*
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [x,x]_P\ \sigma\ [x,x]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [x,x]_P\ \sigma\ [0,2]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,2)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [x,x]_P\ \sigma\ [0,2]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,2)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(0,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ [(x,x)+(2,2)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(0,2)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ [(x,x)+(2,2)]_P$.
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$, *u is even.*
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$, *u is odd.*
	$[\theta x,\phi x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,2)]_P$, *u is odd.*
l	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ u+v\geq 2$.
*	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [0,2]_P\ \sigma\ \{w[\phi,\theta]_P\}$, $u+v\geq 2$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ [x,x]_P$, $u,v\geq 1$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ [0,2]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,2)]_P$, $u,v\geq 1$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ [0,2]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$, $u,v\geq 1$.
*	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ \{w[\phi,\theta]_P\}$, $u+v\geq 2$.
*	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ \{w[\phi,\theta]_P\}$, $u+v\geq 2$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(0,2)]_P$, $u,v\geq 1$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [x,x]_P$, $u,v\geq 1$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,2)]_P$, $u,v\geq 1$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,0)]_P$, $u,v\geq 1$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ [(x,x)+(2,2)]_P$, $u,v\geq 1$.
√	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ [(x,x)+(2,0)]_P$, $u,v\geq 1$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [x,x]_P\ \sigma\ [x,x]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [x,x]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [x,x]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,2)]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [0,2]_P\ \sigma\ [0,2]_P\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(0,2)]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ [(x,x)+(2,0)]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(0,2)]_P$.
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [\theta,\phi]_P\ \sigma\ [(x,x)+(2,2)]_P$.
*	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}$, *u is even,* $u,w\geq 1$.
	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$, $u,v\geq 1$, *u is even.*
√	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,2)]_P$, *u is odd.*
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [\theta x,\phi x]_P$, *u is odd.*
	$[\theta x,\theta x]_S = \{u[2,2]_P$ *or* $v[2,0]_P\}\ \sigma\ [\theta x,\theta x]_P$, $u,v\geq 1$, *u is even.*
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$, *u is even.*
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,0)]_P$, *u is odd.*
	$[\theta x,\theta x]_S = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}\ \sigma\ [(x,x)+(2,2)]_P$, *u is even.*
m *	$[(\theta x,\theta x)+(2,2)] = \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}$, *u is odd,* $w\geq 2$.
√	$[(\theta x,\theta x)+(2,2)] = [(x,x)+(2,0)]_P\ \sigma\ \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}$, *u is odd,* $w\geq 1$.
√	$[(\theta x,\theta x)+(2,2)] = [(x,x)+(2,2)]_P\ \sigma\ \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ \{w[\phi,\theta]_P\}$, *u is even,* $w\geq 1$.
	$[(\theta x,\theta x)+(2,2)] = \{u[\phi,\theta]_P\}\ \sigma\ [(\theta x,\theta x)+(2,2)]$, $u\geq 1$.
	$[(\theta x,\theta x)+(2,2)] = [(x,x)+(2,0)]_P\ \sigma\ \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}$, *u is even.*
	$[(\theta x,\theta x)+(2,2)] = [(x,x)+(2,2)]_P\ \sigma\ \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,2)]_P\ \sigma\ \{w[\phi,\theta]_P\}$, *u is odd.*
	$[(\theta x,\theta x)+(2,2)] = [(x,x)+(2,0)]_P\ \sigma\ \{u[2,2]_P\}\ \sigma\ \{v[2,0]_P\}\ \sigma\ [(x,x)+(2,0)]_P\ \sigma\ \{w[\phi,\theta]_P\}$, *u is odd.*

TABLE 2—*Continued*

n	*	$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ w \geq 1.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ w \geq 1.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma [(\theta x,\theta x)+(2,2)] \sigma \{u[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma [(x,x)+(2,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [0,2]_P \sigma [(\theta x,\phi x)+(2,0)]_P \sigma \{u[\phi,\theta]_P\}.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [x,x]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ w \geq 1.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [x,x]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma [x,x]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [\phi x,\theta x]_P.$
	*	$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma \{u[\phi,\theta]_P\},\ u \geq 1.$
		$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ w \geq 1.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma [(\theta x,\theta x)+(2,2)] \sigma \{u[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma [(x,x)+(2,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [\theta,\phi]_P \sigma [(\theta x,\phi x)+(2,0)]_P \sigma \{u[\phi,\theta]_P\}.$
	√	$[(\phi x,\theta x)+(0,2)]_S = [(x,x)+(0,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ w \geq 1.$
		$[(\phi x,\theta x)+(0,2)]_S = [(\phi x,\theta x)+(0,2)]_P \sigma \{u[\phi,\theta]_P\},\ u \geq 1.$
		$[(\phi x,\theta x)+(0,2)]_S = [(x,x)+(0,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\}.$
		$[(\phi x,\theta x)+(0,2)]_S = [(x,x)+(0,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\}.$
o	*	$[(\theta x,\phi x)+(2,0)]_S = \{u[2,2]_P \text{ or } v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ u,v \geq 1,\ u \text{ is even},\ w \geq 2.$
	√	$[(\theta x,\phi x)+(2,0)]_S = [(x,x)+(2,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ u \text{ is odd},\ w \geq 1.$
	√	$[(\theta x,\phi x)+(2,0)]_S = [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\},\ u \text{ is even},\ w \geq 1.$
		$[(\theta x,\phi x)+(2,0)]_S = \{u[\phi,\theta]_P\} \sigma [(\theta x,\phi x)+(2,0)]_P,\ u \geq 1.$
		$[(\theta x,\phi x)+(2,0)]_S = [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\},\ u \text{ is even}.$
		$[(\theta x,\phi x)+(2,0)]_S = [(x,x)+(2,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\},\ u \text{ is odd}.$
		$[(\theta x,\phi x)+(2,0)]_S = [(x,x)+(2,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\},\ u \text{ is even}.$
p		$[y,y]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma \{w[\phi,\theta]_P\} \sigma [x,x]_P.$
		$[y,y]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma [(\theta x,\theta x)+(2,2)] \sigma \{u[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma [(x,x)+(2,2)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{w[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma [(x,x)+(2,0)]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{w[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma [(\theta x,\phi x)+(2,0)]_P \sigma \{u[\phi,\theta]_P\} \sigma [0,2]_P.$
		$[y,y]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [(x,x)+(2,0)]_P \sigma \{v[2,2]_P\} \sigma \{w[2,0]_P\} \sigma [x,x]_P.$
		$[y,y]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [(x,x)+(2,2)]_P \sigma \{v[2,2]_P\} \sigma \{w[2,0]_P\} \sigma [x,x]_P.$
		$[y,y]_S = [0,2]_P \sigma \{u[\phi,\theta]_P\} \sigma [\phi x,\theta x]_P.$
		$[y,y]_S = [x,x]_P \sigma \{u[2,2]_P\} \sigma \{v[2,0]_P\} \sigma [x,x]_P.$

Using similar arguments for $a\sigma b$, $b\sigma b$, $c\sigma c$, $c\sigma e$, $d\sigma e$, and $e\sigma e$, we have the following lemma.

LEMMA 4.1. *Let N_1 and N_2 be two networks. If $N_1 \sigma N_2$ results in trail covers in more than one class, one in group $T_A \cup T_C \cup T_D \cup T_E$, and the*

others in T_F, then the trail cover in $T_A \cup T_C \cup T_D \cup T_E$ is preferred over the trail covers in T_F.

The rules corresponding to the latter case can be considered redundant. Those corresponding to the former case are called *dominating rules*. This idea can also be extended to the trail cover classes in T_F. By the discussion after Lemma 3.3, a trail cover in T_F with fewer nondistinguished trails is preferred over ones having more nondistinguished trails. For example, we know from Table 1 that $a\sigma f = \{f, k, l\}$. Since f contains only one nondistinguished trail, whereas k and l contain two nondistinguished trails, f is preferred over $\{k, l\}$. To be specific, trail cover classes in T_F excluding p can be distinguished as two sets $S_1 = \{f, g, h, i\}$ and $S_2 = \{j, k, l, m, n, o\}$, based on the number of nondistinguished trails in each trail cover class. For any two networks N_1 and N_2, if $N_1 \sigma N_2$ results in trail covers all in T_F, one in S_1, and the others in S_2, then the resulting trail covers in S_2 are considered to be redundant.

In Table 2, each redundant rule is marked by a $*$ if its graph topology is the same as that of the dominating rule, and by a ✓ otherwise.

5. RECOGNITION OF DEC AND DET NETWORKS

In this section we study the recognition of DEC and DET networks. First, we partition networks into six groups, A, C, D, E, F, and Z. A network N is in A, C, D, and E if there exists a graph representation $(G[N], G[N'])$ that has a trail cover in group T_A, T_C, T_D, and T_E, respectively. A network N is in F if it is not in A, C, D, or E, and there exists a graph representation $(G[N], G[N'])$ that has a trail cover in T_F. All of the remaining networks are said to be in Z. Since trail cover classes a, b, c, d, and e are mutually disjoint, as stated in Lemma 3.2, a network N cannot realize trail covers in class α and in class β for $\alpha, \beta \in \{a, b, c, d, e\}$ and $\alpha \neq \beta$. Therefore, network N cannot be in more than one of the groups A, C, D, E, F, and Z. For example, if N is in A, N cannot be in C, D, E, F, or Z.

Let M be a DET network with child subnetworks $N_1, N_2, \ldots, N_k$ that realizes a DET L in $G[M]$. We assume that M is of type S. To attain results for M of type P, we need only take the duals of the results for M of type S.

A DEC network is a DET network that has a DET (L, L') such that both L and L' are Euler circuits. It is easy to check that the smallest DEC network is the network with a graph representation (G_3, G_3') in Example 2.2. Since there is no trail cover in class $[0, 0]$, the only DEC trail cover must be in $[y, y]$. Let M be a DEC network. M does not contain a child

subnetwork in F, since otherwise L cannot begin and end at the same vertex. Examining the rules listed in Table 2 for $[y, y]_S$, we note that the only rule for $[y, y]_S$ without containing any internal trail is the first rule for p, i.e., $[y, y]_S \leftarrow [0, 2]_P \sigma\{m[\phi, \theta]_P\}\sigma[0, 2]_P$, where $m \geq 0$. Since this rule is the only rule for $[y, y]_S$ to be a DEC, it is a necessary and sufficient condition for DEC networks and we conclude in the following theorem.

THEOREM 5.1. *Let M be a network with type S, and let $N_1, N_2, \ldots, N_k$ be the child subnetworks of M. Then M is a DEC network if and only if exactly two N_i are in T_C and the others are in T_E.*

It follows from Lemmas 3.3, 3.4, and Remark 3.1 that at most two N_i's have $L_G[N_i] \in T_F$, at most two N_i's have $L_G[N_i] \in T_C$, and all of the other N_i's have $L_G[N_i] \in T_A \cup T_E$. Therefore, we have the following lemma in terms of network groups.

LEMMA 5.1. *Assume a network M is a DET network with child subnetworks $N_1, N_2, \ldots, N_k$. Then the following hold*:

(i) *At most two N_i are in F.*

(ii) *At most two N_i are in C.*

(iii) *All of the remaining N_i are in $A \cup E$.*

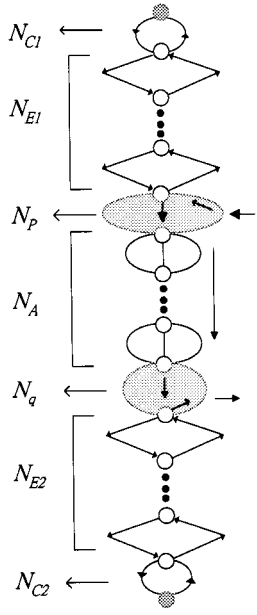
Now we point out some interesting facts about the arrangement of child subnetworks of M in all graph representations $G[M]$ that realize a DET (see Fig. 11 for an illustration):

(i) Child subnetworks in C must be placed at the top and bottom parts of $G[M]$. This follows from the fact that the child subnetwork in C has only one distinguished vertex to concatenate with a trail cover in another child subnetwork.

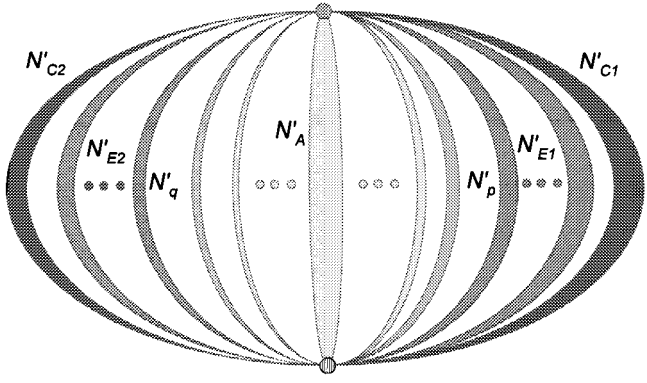
(ii) The subgraph representing child subnetworks in E must be connected to the graph representation of a child subnetwork in C. Therefore, the number of connected components representing child subnetworks in E is no greater than the number of child subnetworks in C.

(iii) Suppose that M contains two child subnetworks N_p and N_q in F. Child subnetworks in E cannot be placed between the subgraphs representing N_p and N_q in G, since otherwise no DET can begin and end with an edge in N_p and an edge in N_q. Furthermore, if M contains child subnetworks in A, those child subnetworks must be placed between N_p and N_q.

LEMMA 5.2. *Let M be a type S network consisting of two child subnetworks N_p, N_q in F, two child subnetworks in C, and some child subnetworks in $A \cup E$. Then M possesses a DET if and only if $[(x, x) + (2, 2)]_P$ or*



Network M



Dual Network M'

FIG. 11. General arrangement of the child subnetworks of a DET network M in all graph representations $(G[M], G[M'])$ that realize a DET.

$[(x, x) + (2, 0)]_P$ *is the trail cover of* N_p *and* N_q. *Furthermore, the resulting DET trail of M is in class* $[y, y]$.

Proof. We can prove the lemma by checking the rules in Table 2. ▮

With this lemma, we know that the converse of Lemma 5.1 is not true. Instead of giving a necessary and sufficient condition for DET networks, we present a linear time algorithm to recognize such networks. To justify the algorithm, we need the following discussion.

Let $X = \{N \in Q \mid |Class(N) \cap \{a, b, c, d, e\}| \geq 1\}$ and $Y = \{N \in Q \mid |Class(N) \cap \{f, g, h, i\}| \geq 1\}$. We have the following lemma.

LEMMA 5.3. *There is no network in* $X \cap Y$.

Proof. By Lemma 3.1, for any $(G[N], G[N'])$ of a network N, the following facts can be proved by induction on the height of the tree structure for N:

(1) If a is in $Class(N)$, then there are exactly two vertices of odd degree in $G[N]$ and $G[N']$, respectively, namely, their two distinguished vertices.

(2) If b is in $Class(N)$, then there are exactly two vertices of odd degree in $G[N]$, namely, its two distinguished vertices, and there is no vertex of odd degree in $G[N']$.

(3) If c is in $Class(N)$, then there is no vertex of odd degree in $G[N]$, and there are exactly two vertices of odd degree in $G[N']$, namely, its two distinguished vertices.

(4) If d or e is in $Class(N)$, then there is no vertex of odd degree in either $G[N]$ or $G[N']$.

It can be observed from the rules in Table 2 that generate f, g, h, and i that for any network N in Y there is a subnetwork (not necessarily a child subnetwork) of N or N' that is a series connection of at least one network containing a trail cover class in $\{[2, 2]_P, [2, 0]_P\}$ and at least one network containing a trail cover class in $\{[0, 2]_P, [\theta, \phi]_P, [\phi, \theta]_P\}$. As a result, if $(G[N], G[N'])$ is any graph representation for some network N in Y, then there exists a nondistinguished vertex which is of odd degree in $G[N]$ or $G[N']$. The lemma is proved. ▮

Using the rules in Table 2, we can develop a linear time algorithm to recognize DET networks. The algorithm works as follows:

Algorithm: RECOG_DET
Input: A network N in tree representation.
Output: "Yes" and resulting trail cover classes if N possesses a DET, and "No" otherwise.

Method: For each nonleaf node x, use $N(x)$ to denote the subnetwork that it represents and $N(x)'$ to denote the dual of $N(x)$. Define $M(x)$ to be one of the networks $N(x)$ and $N(x)'$ that is of type S. Moreover, $M(x)'$ denotes the dual of $M(x)$. To be precise, we define $M(x) = N(x)$ if $N(x)$ is a type S network, and $M(x) = N(x)'$ otherwise. From leaves to the root, level by level, calculate $Class(M(x)) \subseteq TC$, which denotes the set of trail cover classes realized on $M(x)$, by the following steps.

1. If x is a leaf node, $Class(M(x)') = \{[2, 2]_P\}$.

2. If x is a nonleaf node, compute $Class(M(x))$ using the rules in Table 2.

3. If none of the rules can be applied, then output "No" and STOP.

4. If x is the root and one of $[2, 2]_S$, $[2, 0]_S$, $[0, 2]_S$, $[x, x]_S$, or $[y, y]_S$ is in $Class(M(x))$, then output "Yes" and STOP. If x is the root and none of the above five classes is in $Class(M(x))$, then output "No" and STOP.

5. If x is a nonleaf node, calculate $Class(M(x)')$ by setting $Class(M(x)') =$ dual of $Class(M(x))$, and proceed to next node.

The correctness of RECOG_DET follows from our rules. We observe that the time spent on Step 2 is crucial in determining the time complexity of our algorithm. To discuss the time spent on Step 2, we describe the method used to implement this step.

Assume that $M(x)$ has w child subnetworks $M'(x_1), M'(x_2), \ldots, M'(x_w)$. Let K be any trail cover in $Possible(M(x))$. Observe that any trail cover in $\{f, g, h, i\}$ contains exactly one distinguished trail, whereas any trail cover in $\{j, k, l, m, n, o\}$ contains exactly two nondistinguished trails. Moreover, a nondistinguished trail can be concatenated with at most one other trail. Hence all of the trail covers for $M'(x_i)$ induced by K satisfy one of the following three cases: Case (1) in all $M'(x_i)$, every induced trail cover is in $\{a, b, c, d, e\}$; Case (2) in all $M'(x_i)$, at most two induced trail covers are in $\{f, g, h, i\}$, and the remaining in $\{a, b, c, d, e\}$; Case (3) in all $M'(x_i)$, only one induced trail cover is in $\{j, k, l, m, n, o\}$, and those remaining are in $\{a,b, c, d, e\}$.

Let $a(M(x)) = |\{i \mid a \in Class(M'(x_i))\}|$. Similarly, we define $b(M(x))$, $c(M(x))$, $d(M(x))$, and $e(M(x))$. We can compute $a(M(x))$, $b(M(x))$, $c(M(x))$, $d(M(x))$, and $e(M(x))$ in $O(w)$ time. We first discuss the trail cover classes in $Class(M(x))$ for trail covers in Case (1). Since $|Class(M'(x_i)) \cap \{a, b, c, d, e\}| \neq 0$ for every i, it follows from Lemma 3.2 that each child subnetwork N_i has a unique trail cover class in $\{a, b, c, d, e\}$. To find all of the trail cover classes for the trail cover of $M(x)$ in Case (1), we scan all of the rules in Table 2 that are a series connection of $a(M(x))$ trail covers in class a, $b(M(x))$ trail covers in class b, $c(M(x))$ trail covers

in class c, $d(M(x))$ trail covers in class d, and $e(M(x))$ trail covers in class e. Although Table 2 has many rules, the number of rules is still bounded by a constant. Thus we use additional $O(1)$ time to find all of the trail cover classes in Case (1) once we have $a(M(x))$, $b(M(x))$, $c(M(x))$, $d(M(x))$, and $e(M(x))$.

Consider Case (2). It follows from Lemma 5.3 that there is no network in $X \cap Y$. We can easily find all child subnetworks with an induced trail cover in $\{f, g, h, i\}$. Moreover, there are at most two such child subnetworks. We assume without loss of generality that the induced trail covers of $M'(x_1)$ and $M'(x_2)$ are in $\{f, g, h, i\}$ and the remaining child subnetworks are in X. Since there are at most four possible trail cover classes in $\{f, g, h, i\}$, we scan the rules 16 times at most to find all trail cover classes in Case (2). Additional $O(1)$ time will be used to find all of the trail cover classes for $M(x)$.

Finally, in Case (3), assume that $M'(x_1)$ has an induced trail cover K_1 in $\{j, k, l, m, n, o\}$. It follows that exactly one of the following three statements holds: (i) $|Class(M'(x_1)) \cap \{a, b, c, d, e\}| = 1$; (ii) $|Class(M'(x_1)) \cap \{f, g, h, i\}| \geq 1$; or (iii) $Class(M'(x_1)) \subset \{j, k, l, m, n, o\}$. When statement (i) holds, we assume that $a \in Class(M'(x_1))$ and K_1 is in class j. Then any trail cover K for $M(x)$ that includes K_1 as an induced trail cover must induce some trail covers in $\{a, b, c, d, e\}$ for other child subnetworks. That is, all of the other child subnetworks must be in X. Thus K is a series connection of a trail cover in class j, $a(M(x)) - 1$ trail covers in class a, $b(M(x))$ trail covers in class b, $c(M(x))$ trail covers in class c, $d(M(x))$ trail covers in class d, and $e(M(x))$ trail covers in class e. We can find the trail cover class for K in $O(1)$ time. Since there are only six trail cover classes in $\{j, k, l, m, n, o\}$, we may use $O(1)$ time to find all of the trail cover classes that are a series connection of a trail cover in $\{j, k, l, m, n, o\}$ for $M'(x_1)$ and some other trail covers in $\{a, b, c, d, e\}$ for $M'(x_i)$ with $i \neq 1$. Moreover, if $a \notin Class(M'(x_1))$, exactly one of b, c, d, and e can be in $Class(M'(x_1))$. The discussion for these situations is analogous, so statements (ii) and (iii) can be treated similarly. Since the possible induced trail covers in $\{j, k, l, m, n, o\}$ may be in $M'(x_i)$ for $1 \leq i \leq w$, we can find all such trail cover classes in $O(w)$ time.

From leaves to the root and level by level, the time spent on Step 2 for each node x is bounded by $O(d_x)$, where d_x is the number of children for x. Therefore, we have the following theorem.

THEOREM 5.2. *Algorithm RECOG_DET recognizes DET networks in linear time.*

From the above discussion, it is clear that the time complexity of our algorithm is $T(n) = \alpha n$, where α is a large constant. α can be viewed as the number of rules in Table 2.

6. AN EXAMPLE

To illustrate the above algorithm, we use the network given by the Boolean function $((((((\mathbf{a} \wedge \mathbf{b}) \vee \mathbf{c}) \wedge (\mathbf{f} \vee \mathbf{g})) \vee (\mathbf{e} \wedge \mathbf{d})) \wedge \mathbf{h}) \vee (\mathbf{k} \wedge \mathbf{l})) \wedge (\mathbf{i} \vee \mathbf{j}) \wedge (\mathbf{m} \vee \mathbf{n})$, which has two pairs of graph representations, shown in Figure 12. The trail covers realized in a parallel type subnetwork are obtained by taking the duals of the trail covers derived from the rules for series connection. To obtain the trail covers realized in a series type subnetwork, we need to take the duals of the trail covers in its child subnetworks and then apply the rules in Table 2. Figure 13 also illustrates this procedure. The network has two nonequivalent DET's **dehijklmnabfgc** and **abcdefghijklmn**, which satisfy the fourth and the ninth rules for $[y, y]_S$. From this final result, we can determine the trail covers realized in

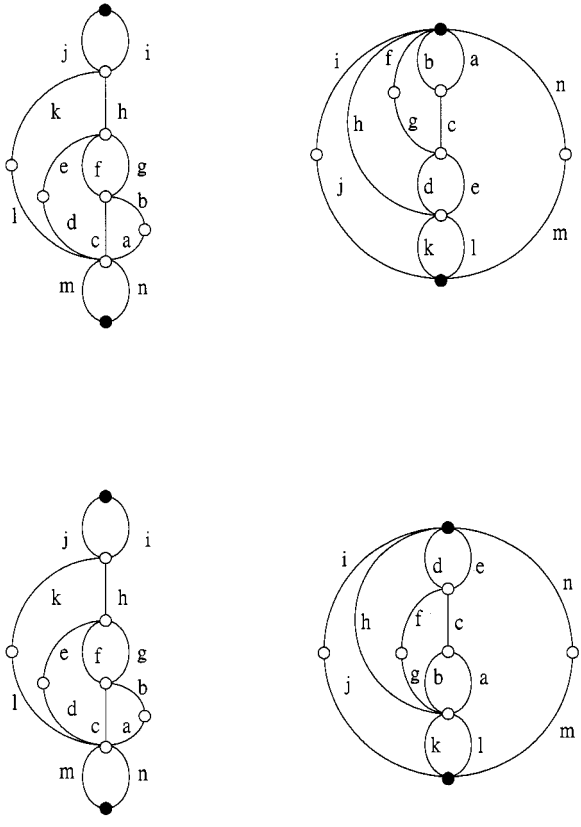


FIG. 12. Two nonisomorphic graph pairs of a DET network realizing different DETs.

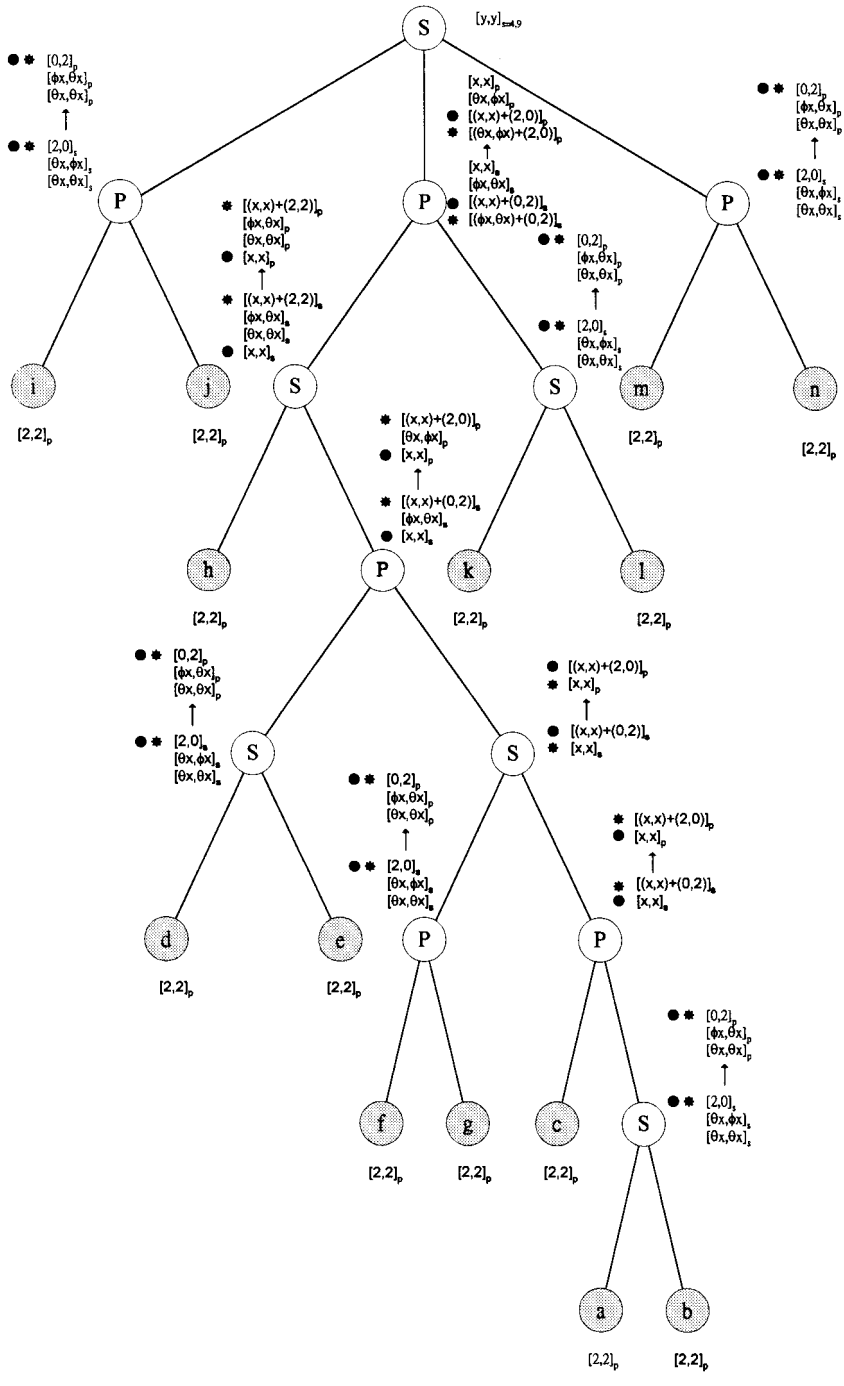


FIG. 13. Illustration of the RECOG_DET algorithm.

each subnetwork, from the root to leaves, that result in DETs. The ones that lead to the DET satisfying the fourth rule for $[y, y]_S$ are marked by a $\bullet$, and those leading to the ninth rule are marked by a $*$.

7. CONCLUSION

In this paper we study the DET problem that arises from VLSI layout. We build a multivalued function (Table 1) to discuss the concatenation of these trail cover classes. It is observed that Table 1 is closed under series connection, parallel connection, and taking the dual. There are exactly 16 trail cover classes. To discuss the possible concatenation of all permutations of trail cover classes, we study the basic structures of the trail cover classes of DET networks. Then we use a program to generate rules for trail cover class composition. Using these rules, we can build a linear time algorithm to recognize those networks that possesses DET trails. The approach that uses a program to generate rules is very interesting. We believe that such an approach can be used in algorithm designs for other problems, especially those complicated problems using dynamic programming. We also believe that the original problem that computes $DCT(N)$ for any network N might be a candidate problem using this approach. Actually, we have tried this approach on $DCT(N)$. However, we need to reclassify those trail cover classes so that it is closed under series connection, parallel connection, and taking the dual. We believe that at least 540 trail cover classes should be considered as we analyze the behavior of $DCT(N)$. Because the problem is so complicated, we still cannot solve the DCT problem.

We also want to point out that RECOG_DET can be translated into a parallel algorithm. Tree contraction [1] is a general technique for designing parallel algorithms for certain problems defined on a tree. The DET problem is also a problem defined on a tree. It is easy, but not trivial, to modify the technique of tree contraction to solve the DET problem in $O(\log n)$ time with $O(n/\log n)$ processors under EREW PRAM [18].

In VLSI layouts, circuit designers may wish to keep the DETs to pursue height or floor plan optimization. We have seen in Figure 12 that a DET network may realize various DETs in different pairs of graph representations. Figure 14 shows the corresponding geometric layouts. The layout designers may prefer the layout with the smaller height or the layout that fits his floor plan. For this reason, it is usful to design a data structure that keeps the DETs in all graph representations for any network [14].

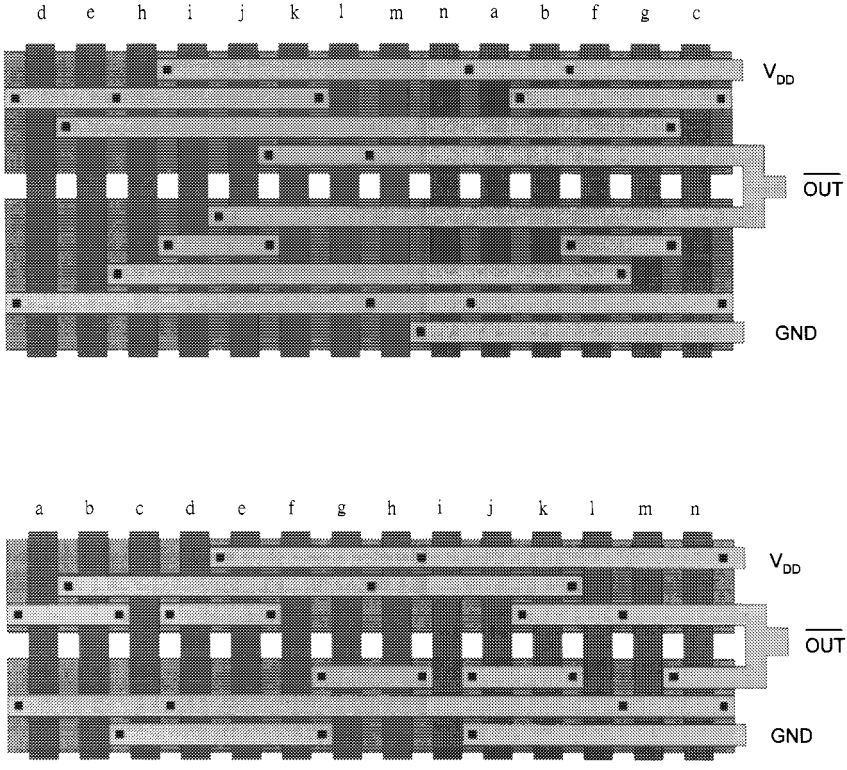


FIG. 14. Two different geometric layouts for the DET network in Figure 13.

ACKNOWLEDGMENTS

The authors are very grateful to the anonymous referees for their thorough review of the paper and many concrete and helpful suggestions. This work was supported in part by the National Science Council of the Republic of China under contract NSC83-0208-M009-034.

REFERENCES

1. K. Abrahamson, D. Dadoun, D. G. Kirkpatrick, and T. Przytycka, A simple parallel tree contraction algorithm, *J. Algorithms* **10** (1989), 287–302.
2. M. W. Bern, E. L. Lawler, and A. L. Wong, Linear time computation of optimal subgraphs of decomposable graphs, *J. Algorithms* **8** (1987), 216–235.
3. J. A. Bondy and U. S. R. Murty, ''Graph Theory with Applications,'' Elsevier, New York, 1976.
4. B. S. Carlson, C. Y. Chen and D. S. Meliketian, Daul Eulerian properties of plane multigraphs, *SIAM J. Disc. Math.* **8** (1995), 33–50.

5. R. J. Duffin, Topology of series parallel networks, *J. Math. Anal. Appl.* **10** (1965), 303–318.
6. L. H. Hsu, J. Y. Hwang, T. Y. Ho, and C. H. Tsai, A linear time algorithm to recognize the double Euler trail for series-parallel networks, *in* "Lecture Notes in Computer Science, No. 557, ISA '91 Algorithms" (W. L. Hsu and R. C. T. Lee, Eds.), pp. 316–325, Springer-Verlag, Berlin, 1991.
7. L. H. Hsu and S. Y. Wang, Maximum independent number for series-parallel networks, *Networks* **21** (1991), 457–468.
8. T. Kikuno, N. Yoshida, and Y. Kakuda, A linear time algorithm for domination number of a series parallel networks, *Discrete Appl. Math*. **5** (1983), 299–311.
9. T. Lengauer and R. Müller, Linear algorithm for optimizing the layout of dynamic CMOS cells, *IEEE Trans. Circuits Systems* **35** (1988), 279–285.
10. R. Maziasz and J. Hayes, Layout optimization of CMOS functional cells, *in* "Proceedings of the 24th ACM/IEEE Design Automation Conference," 1987, pp. 544–551.
11. R. Maziasz and J. Hayes, Layout optimization of static CMOS functional cells, *IEEE Trans. Computer-Aided Design* **9** (1990), 708–719.
12. R. Nair, A. Bruss, and J. Reif, Linear time algorithms for optimal CMOS layout, *in* "VLSI: Algorithms and Architecture" (P. Bertolazzi and F. Luccio, Eds.), pp. 237–338, North-Holland, Amsterdam, 1985.
13. J. Riordan and C. E. Shannon, The number of two terminal series parallel networks, *J. Math. Phys*. **21** (1942), 83–93.
14. T. Y. Sung, L. H. Hsu, and J. Y. Hwang, Data structure for graph representations of a network having double Euler trails, in "Information Processing 92, Volume I: Algorithms, Software, Architecture" (J. van Leeuwen, Ed.), pp. 436–442, Elsevier Science Publishers B.V. (North-Holland), Amsterdam, 1992.
15. K. Takamizawa, T. Nishizeki, and N. Saito, Linear time computability of combinatorial problems on series parallel graphs, *J. Assoc. Comput. Mach*. **29** (1982), 623–641.
16. T. Uehara and C. vanCleemput, Optimal layout of CMOS functional arrays, *IEEE Trans. Comput*. **30** (1981), 305–312.
17. J. Valdes, R. E. Tarjan, and E. L. Lawler, The recognition of series parallel digraphs, *SIAM J. Comput*. **11** (1982), 289–313.
18. J. J. Wang, S. Y. Wang, L. H. Hsu, and T. Y. Sung, Extended tree contraction for some parallel algorithms on series-parallel networks, *J. Inform. Sci. Engrg*. **13** (1997), 665–680.