Formal Analysis of Electrical Circuit Network Topologies using Theorem Proving

Kubra Aksoy
Department of Electrical and Computer Engineering
Concordia University
Montreal, Quebec, Canada
k_aksoy@ece.concordia.ca

Adnan Rashid
Department of Electrical and Computer Engineering
Concordia University
Montreal, Quebec, Canada
rashid@ece.concordia.ca

Osman Hasan
School of Electrical Engineering and Computer Science
National University of Science and Technology
Islamabad, Pakistan
osman.hasan@seecs.nust.edu.pk

Sofiène Tahar
Department of Electrical and Computer Engineering
Concordia University
Montreal, Quebec, Canada
tahar@ece.concordia.ca

2025 IEEE International systems Conference (SysCon) | 979-8-3315-0818-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/SYSCON64521.2025.11014814

***Abstract*—Electrical circuit networks consist of various electrical components connected in different arrangements that are used to process electrical signals, such as voltages and currents. Moreover, they are widely used in many engineering and physical systems, such as control, embedded and communication systems. In this work, we propose to use higher-order-logic theorem proving for formally analyzing electrical circuit networks using network topology matrices. In particular, we formalize in the Isabelle/HOL theorem prover incidence and circuit (loop) matrices, which are widely used topology matrices. Our formalization is based on modeling the system as a network using the notion of directed graphs that are algebraically represented by matrices. Next, we use this formalization to formally analyze various network topologies, such as Δ-Y and Π-L topologies, where we perform Kirchhoff's current and voltage laws based analysis in a generic way.**

***Index Terms*—Electrical Network Topology, Matrix Theory, Graph Theory, Theorem Proving, Isabelle/HOL**

I. Introduction

Electrical circuit networks are composed of various electrical components, such as resistors, capacitors, inductors and current/voltage sources, and are used to process electrical signals, such as voltages and currents. They can take different forms/configurations based on a specific arrangement of these network components and their interconnection. Some of these well-known topologies are Δ, Y, Δ-Y and Π-L topologies [1]. Moreover, these electrical circuits are considered as an integral part of many engineering and physical systems, such as control systems (e.g., to design controllers), embedded systems (e.g., to design integrated networks) and communication systems (e.g., to design analog to digital converters). Due to their usage in safety-critical applications, their accurate analysis is of utmost importance.

The analysis of an electrical network is mainly based on constructing its equivalent mathematical model by using system governing laws. In particular, we need to model the currents and voltages passing through the electrical components and their interactions in the corresponding electrical network using the Kirchhoff's laws, such as Kirchhoff's current law (KCL) and Kirchhoff's voltage law (KVL) [2]. In this regard, we need to apply KCL and KVL at each node of a network to obtain a set of equations providing a relationship of voltages and currents across its various components. However, modeling and solving these equations easily become cumbersome when dealing with complex electrical circuit networks. In addition, there is also a chance of missing some nodes or electrical components during this kind of analysis. Therefore, we need some alternate approach that can be used to analyze electrical circuit networks.

Network topology [3] enables the depiction of relationships among different components in an electrical network through the use of graphs consisting of nodes and edges, and can be considered as an alternate to the above-mentioned approach. Here, edges of the graph represent the direction of the flow of current through electrical components and nodes capture their interconnections in an electrical network. Moreover, linear algebraic methods, particularly matrix algebra [4], form one basis for analyzing these electrical networks, where the graph representations of these systems are transformed into various matrix-based representations. Matrices allow representing large and complex graphs more compactly and efficiently than a visual representation, which further motivates the need for algebraic foundational development in network theory. Some of the widely used topological matrices are the incidence and the circuit (loop) matrices [3]. For instance, the incidence matrix provides a relationship between the nodes and edges of the systems, while the circuit matrix shows the relationship between circuits and edges. By analyzing these matrices, we can verify the accuracy of the system model based on the topological characteristics such as the number of nodes, edges, and closed-loops. Furthermore, we only need to analyze matrix-based representations rather than solving a set of equations providing a relationship of voltages and currents. Therefore, this approach is quite effective and efficient particularly for large electrical networks.

In this work, we propose to use higher-order-logic (HOL) theorem proving for formally analyzing electrical circuit networks using the network topology matrices. In particular, we formalize the incidence and the circuit matrices based on the network modeled as a directed graph in Isabelle/HOL [5]. The main motivation for considering directed graph-based models

is their generic nature compared to undirected graphs. Moreover, directed graphs are suitable for representing the network topologies, where the directed edges capture the direction of the current flowing through an electrical component [6]. Next, we use our formalization of the network topology matrices for formally analyzing various network topologies, such as Δ-Y and Π-L with an aim to construct a comprehensive library of formal electrical circuit network analysis in Isabelle/HOL. We start our proposed formalization by modeling the network incidence system based on the incidence structure between nodes and edges of the directed graph representation, which is further used to formalize the incidence matrix and to formally verify its classical properties in Isabelle/HOL. Next, we formalize the network circuit system on top of the network incidence system to model circuit matrices and formally verify related properties. Moreover, we formalize the generic representations of KVL and KCL, which can be used to analyze an electrical circuit network of any size. Finally, we use our formalizations of the network topology matrices, and KVL and KCL to formally analyze various electrical circuit network topologies using Isabelle/HOL. One of the primary reasons for choosing Isabelle/HOL for the proposed formalization is the availability of extensive reasoning support for graphs and matrices.

The rest of the paper is structured as follows: Section II overviews the related work regarding the formalization of networks and graphs. We provide formalizations of incidence and circuit matrices in Sections III and IV. Section V presents the formal analysis of various network topologies using Isabelle/HOL. Section VI provides some discussion while highlighting the usefulness of the proposed approach. We conclude the paper in Section VII with some future directions.

II. Related Work

HOL theorem proving has been used for formally analyzing the electrical networks. For example, Zaki et al. [7] used the HOL4 theorem prover for formally verifying the continuous models of analog circuits. In particular, the authors developed a basic library of analog circuit analysis in HOL4, where they have been able to successfully apply their proposed framework to formally analyze an RLC (resistor, inductor and capacitor) network and a delta-sigma modulator. Similarly, Taqdees et al. [8] used the HOL Light theorem prover for formally verifying the transfer functions of linear analog filters, such as first and second-order Sallen-Key low-pass filters. Later, Rashid et al. [9] formally analyzed a 4-π soft error crosstalk model using HOL Light, which is widely used in integrated circuits. More recently, Rashid et al. [10] proposed a framework, FASiM, which enable an automatic formal analysis of the Simulink models of linear analog circuits using HOL Light. However, all these research contributions are able to analyze a certain set of electrical circuits only and do not provide any verification of the electrical networks topologies.

Networks and graphs have also been formalized in many HOL theorem provers, such as Isabelle/HOL, Coq, Lean and Mizar. For instance, Noschinski [11] formally developed a general graph theory emphasizing on the basics of directed graphs using Isabelle/HOL structures such as records. Similarly, Lammich et al. [12] proposed another directed graph formalization based on sets and formally verified various network flow algorithms in Isabelle/HOL. However, these formalizations mainly focus on either the general concept of networks and graphs without considering incidence structures/topology matrices or verifying network flow algorithms.

Similarly, there are only a few formalizations of incidence matrices available in the literature that represent combinatorial structures, such as *designs* and *graphs* [13]. For instance, Heras et al. [14] formalized the incidence matrix using the Coq/SSReflect theorem prover and used it to formally analyze 2D digital image processing systems. More recently, Edmonds et al. [15] used Isabelle/HOL to formalize Fisher's inequality, which is based on developing incidence matrices over rings as a linear algebraic representation of designs, using the linear algebraic proof approach [16]. However, both of the preceding contributions only provide the formalization of simple incidence matrices having entries as 0 and 1 only, i.e, modeling undirected graphs under some conditions. Moreover, these contributions do not consider directed graphs for the formalization of the incidence matrix, which incorporates the direction aspect and considers the entries of the matrix as 0, 1 and -1. Also, the network topology matrices are not used for formally analyzing electrical networks topologies, which is the main scope of the current paper.

III. Formalization of Incidence Matrices

A. Incidence Matrices

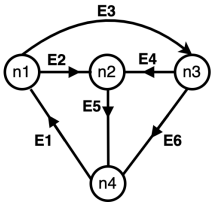
Incidence matrices provide algebraic representations of networks captured by directed graphs and describe the relationship between nodes and edges through the incidence structures. They are mathematically defined as follows [17]:

Definition 1. *Incidence Matrix of a Network*
Consider a network with m nodes, n edges, and no self-loops. An incidence matrix $\mathcal{A} = [a_{ij}]$ is defined as an $m \times n$ matrix for $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, n\}$, such that

$$a_{ij} = \begin{cases} 1 & \text{if edge } j \text{ is positively incident with node } i \\ -1 & \text{if edge } j \text{ is negatively incident with node } i \\ 0 & \text{otherwise} \end{cases}$$

where an edge is said to be positively incident with a node if the direction of the edge is away from the node. Similarly, an edge is negatively incident with a node if the edge is directed to the node.

Figure 1 depicts a directed graph corresponding to a network, and its incidence matrix representation. The graph in Figure 1(a) has 4 nodes and 6 edges, where every edge is represented by a pair of nodes. The entries in the matrix are determined by the incidence relationship between nodes and edges. For instance, $edge\ E5$ is represented by ($n2$,$n4$), which is positively incident with $node$ 2 and negatively incident with $node$ 4. The matrix elements for this relation are given as $a_{25} = 1$ and $a_{45} = -1$ as shown in Figure 1(b).



(a)

	E1	E2	E3	E4	E5	E6
n1	−1	1	1	0	0	0
n2	0	−1	0	−1	1	0
n3	0	0	−1	1	0	1
n4	1	0	0	0	−1	−1

(b)

Fig. 1: (a) Directed Graph Representation of a Network; (b) Incidence Matrix [17]

B. HOL Formalization of Incidence Matrices

We use Isabelle's module system, locales [18], to model directed graph-based network systems, and to construct equivalent matrix representations. Locales are often used for algebraic structures with parametrization. They can also be extended hierarchically, allowing us to combine existing locales into a new locale with new parameters and assumptions. To formalize incidence matrices, we first formally define a network system as a *locale* module comprising the main components of a network: a list of nodes is formalized as a type of list, a list of edges is formalized as a pair list, and their relationship as well-formed assumptions[1]. The *locale* ensures that all edges consist of distinct nodes and the network has no self-loops and multi-edges. Here, we use lists instead of sets for the formalization of the network system since we need an ordering structure that allows us to efficiently define a relationship of edges with their respective nodes.

```
locale network_system =
fixes nodes_list :: 'a nodes (𝒩s)
and edges_list :: 'a edges (ℰs)
assumes network_wf: e ∈ set ℰs ⟹ fst e ∈ set 𝒩s ∧
                                  snd e ∈ set 𝒩s ∧ fst e ≠ snd e
assumes distincts: distinct 𝒩s distinct ℰs
```

where the function `set` accepts a list and returns a set. Similarly, the function `distinct` takes a list and ensures that elements of the list are disjoint. Furthermore, the assumption `network_wf` ensures that the network has no self-loop.

To obtain a valid network system that can be converted into its algebraic representation, we define a new locale with non-empty edge lists. This locale is combined with `finite_netw_sys` *locale* built on top of `network_system` *locale*, which assumes that the sets of edges and nodes are finite.

```
locale nonempty_network_system = finite_netw_sys +
 assumes edges_nempty: ℰs ≠ []
```

Next, we need to define the positive and negative incidence by incorporating the direction of edges. To formalize the positive incidence, we first define a set $\mathtt{I_P}$, whose elements are pairs of nodes and edges, with the first element being a node and the second element its edge directed away from the node.

[1]We abbreviate 'a as 'a node, 'a list as 'a nodes, 'a × 'a as 'a edge, and ('a × 'a) list as 'a edges for a better readability.

```
definition I_P ≡ {(v,e). e ∈ set ℰs ∧ fst e = v}
```

Thereafter, we formalize a predicate `pos_incident`, which ensures that the node `v` is positively incident to the edge `e` if their pair is in the set $\mathtt{I_P}$.

```
definition pos_incident v e ≡ (v,e) ∈ I_P
```

Similarly, we formalize negative incidence using a set $\mathtt{I_N}$ as:

```
definition I_N ≡ {(v,e). e ∈ set ℰs ∧ snd e = v}
```

```
definition neg_incident v e ≡ (v,e) ∈ I_N
```

We now formalize an incidence matrix (Definition 1) as:

```
definition incidence_matrix ::
'a nodes ⇒ 'a edges ⇒ ('b :: field_char_0 mat) where
 incidence_matrix Ns Es ≡
  mat (length Ns) (length Es) (λ(i,j).
   if (fst (Es!j) ≠ snd (Es!j) ∧
      (Ns!i) = fst (Es!j)) then 1 else
   if (fst (Es!j) ≠ snd (Es!j) ∧
      (Ns!i) = snd (Es!j)) then -1 else 0)
```

where the function `length` accepts a list and provides the number of elements of the list. Moreover, the function `incidence_matrix` generates a matrix providing relationships between all nodes and edges of the directed graph. Note that this function is defined outside the locale to be equivalent for an arbitrary network system consisting of a list of nodes and edges. Therefore, the condition `fst (Es!j) ≠ snd (Es!j)` is added to ensure that the definition is free of self-loops. Moreover, the axiomatic type `field_char_0` is chosen for the matrix elements. This type can easily be rendered into the real field, where the incidence matrix can be algebraically defined over the real field [19]. Additionally, the type's properties like $1 \neq -1$ allow an easy use for proving lemmas and efficient computations. More details about the incidence matrix formalization and properties verified in Isabelle/HOL can be found in [20].

IV. Formalization of Circuit Matrices

A. Circuit Matrices

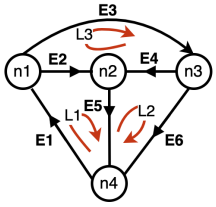
For a directed graph with l circuits and n edges, a circuit matrix is mathematically expressed as [17]:

Definition 2. *Circuit Matrix of a Network*
Consider a network with m nodes, n edges, and l circuits. The circuit matrix $\mathcal{C} = [c_{kj}]$ is an $l \times n$ matrix, where $k \in \{1, 2, \ldots, l\}$ and $j \in \{1, 2, \ldots, n\}$, such that

$$c_{kj} = \begin{cases} 1 & \text{if edge } j \text{ is in the circuit } k, \text{ and the direction of the edge } j \text{ and the circuit } k \text{ are the same} \\ -1 & \text{if edge } j \text{ is in the circuit } k, \text{ and the direction of the edge } j \text{ and the circuit } k \text{ are opposite} \\ 0 & \text{if edge } j \text{ is not in the circuit } k \end{cases}$$

where a circuit of a directed graph is defined as a closed path represented by a finite sequence of distinct edges captured as pairs of nodes as $[(n_1, n_2), (n_2, n_3), \ldots, (n_{i-1}, n_i)]$ for $n \geq 2$ such that $n_1 = n_i$.

Figure 2 depicts a directed graph of a network with some arbitrary circuits and its corresponding circuit matrix. Here, the circuit matrix takes the set of circuits and the set all edges as inputs, where each row of the circuit matrix corresponds to the circuit in the graph, and each column represents an edge in the graph. For $L2 = [-E4,\ E6,\ -E5]$[2], the first element of the circuit list, $-E4$, indicates that edge $E4$ is in the circuit and its direction is opposite to the circuit's orientation. This is reflected in the matrix as $c_{24} = -1$, as depicted in Figure 2(b). It is important to note that all circuits can have two directions, either clockwise or counterclockwise, and the direction of circuits are independent of the direction of edges.



(a)

	E1	E2	E3	E4	E5	E6
L1	1	1	0	0	1	0
L2	0	0	0	−1	−1	1
L3	0	−1	1	1	0	0

(b)

Fig. 2: (a) Some circuits in Directed Graph of Figure 1(a); (b) Circuit Matrix

B. HOL Formalization of Circuit Matrices

To formalize a circuit matrix, we need to model a circuit and incorporate it into the network system. A circuit of a directed graph is formalized as follows:

```
definition circuit :: 'a edges ⇒ bool where
circuit ps ≡ (is_gen_closed_path (fst (hd ps))
                    ps (fst (hd ps))) ∧ length ps ≥ 2
```

where `is_gen_closed_path` represents a distinct pair list of closed train (walk). This is defined as a sequence of edges where the first element of the head edge is the same as the second element of the last edge in the list, and is obtained by using the function `pred_netw_dedge_seq` that is formalized as:

```
fun pred_netw_dedge_seq ::
      'a node ⇒ 'a edges ⇒ 'a node ⇒ bool  where
pred_netw_dedge_seq v1 [] vx = (v1 = vx)  |
pred_netw_dedge_seq v1 (p1 # ps) vx =
                  (v1 = fst p1 ∧ fst p1 ≠ snd p1 ∧
 pred_netw_dedge_seq (snd p1) ps vx)
```

In the next step, we incorporate the concept of the circuit into the network system by defining a new locale, as follows:

```
locale circuit_system = nonempty_network_system +
 fixes circuits_list :: 'a edges list (ℒs)
 assumes wellformed_1: ls ∈ set ℒs ⟹
             set ls ⊆ symcl ℰ ∧ length ls ≤ length ℰs
 assumes wellformed_2: ls ∈ set ℒs ⟹
                      circuit ls ∧ circuit (reverse ls)
 and distinct: distinct ℒs
```

where `symcl` accepts a Cartesian set and guarantees that this set contains elements with their reversed version. The assumption `wellformed_1` ensures that every element of the list of circuits $\mathcal{L}$s is a subset of the symmetric relation of edges and satisfies the condition that the size of the circuit cannot be larger than the number of edges in the graph. Similarly, `wellformed_2` guarantees that every element of the circuit list is a circuit and its reverse is also a circuit. The assumption `distinct` ensures the non-repetition of circuits in the circuit list. Additionally, we formally define a `nonempty_circuit_system` locale on top of the `circuit_system`, which ensures that the circuit system has at least one circuit in the circuit list.

```
locale nonempty_circuit_system = circuit_system +
      assumes circuits_list_nempty: ℒs ≠ []
```

Now, it is necessary to describe the directional relation between circuits and the corresponding edges of the given network. The circuit is said to be positively oriented with an edge if the edge is in the circuit, while the circuit is negatively oriented with the edge if the edge is in the reversed circuit. We develop these definitions by explicitly indicating the direction of the circuits while assuming each edge is positively directed. For instance, the set of pairs whose elements are edges and circuits, abbreviated as L_p, is used to designate the relation between edges and circuits where edges are in positively oriented circuits.

```
definition Lp ≡ {(x,es). x ∈ set ℰs ∧ es ∈ set ℒs
                              ∧ x ∈ set (circuit_list es)}
```

Similarly, we formalize a predicate for the relationship indicating that an edge is in a negatively oriented circuit.

```
definition in_neg_circuit x es ≡ (x,es) ∈ Ln
```

where L_n is a set of pairs consisting of the edge and the circuit, formally defined as follows:

```
definition Ln ≡ {(x,es). x ∈ set ℰs ∧ es ∈ set ℒs
                    ∧ x ∉ set (circuit_list es)
                     ∧ x ∈ set (reverse (circuit_list es))}
```

Here, the function `reverse` takes a pair list and reverses its order by swapping the elements of each pair in the list. Therefore, we obtain the concept of negatively oriented circuits using the function `reverse`. It is important to highlight that we assumed each circuit list as a subset of the set of edges with symmetric relation in the circuit system environment. By doing so, we are able to work with networks that have both directed and semi-directed circuits [19]. Note that a semi-directed circuit in a directed graph is a circuit in the corresponding undirected graph. Now, the formalization of the circuit matrix is given as follows:

```
definition circuit_matrix::
 'a edges list ⇒ 'a edges ⇒ ('b :: field_char_0 mat)
 where circuit_matrix Ls Es ≡
       mat (length Ls) (length Es) (λ(k,j).
 if (Es!j) ∈ set (circuit_list (Ls!k)) then 1 else
     if (Es!j) ∈ set (reverse (circuit_list (Ls!k)))
          then -1 else 0)
```

[2]The negative sign indicates the opposite direction of the edges with respect to the orientation of the circuit, e.g., $-E4 = (n2, n3)$.

More details about the formalization of circuit matrices and the verification of various classical properties in Isabelle/HOL can be found in [20].

V. Formal Analysis of Electrical Circuit Network Topologies

We now use the formalization of the network topology matrices, presented in Sections III and IV, for formally analyzing various electrical circuit network topologies, such as Δ-Y and Π-L topologies. Other sorts of topologies, such as the H, Box or Bridge topologies [21], can be formalized in the same way.

A. Formal Analysis of the Δ-Y Network Topology

A Δ-Y network topology consists of a Δ network connected in parallel to a Y network [21]. Figure 3(a) depicts a Δ network topology, which consists of three components $C1$, $C3$ and $C6$, shown as rectangular boxes. These boxes can represent any (one or more) of the electrical components, i.e., resistors, capacitors, inductors or current/voltage sources. For example, for a choice of $C1$ as a current source, $C3$ as a resistor and $C6$ as an inductor, the corresponding topology results into an RL (resistor-inductor) circuit. Similarly, Figure 3(b) depicts a Y circuit topology comprising of three components $C2$, $C4$ and $C5$, which can be any of the electrical components. Moreover, a parallel connection of these topologies represents a Δ-Y network topology, which is depicted in Figure 3(c).

We start the analysis of the Δ-Y network by applying KCL on nodes $n1$, $n2$, $n3$ and $n4$ (Figure 3(c)), which results into the following set of equations:

$$\begin{aligned} &\text{Node n1} : i_2(t) + i_3(t) = i_1(t) \\ &\text{Node n2} : i_2(t) + i_4(t) = i_5(t) \\ &\text{Node n3} : i_4(t) + i_6(t) = i_3(t) \\ &\text{Node n4} : i_5(t) + i_6(t) = i_1(t) \end{aligned} \tag{1}$$

where $i_1(t)$, $i_2(t)$, $i_3(t)$, $i_4(t)$, $i_5(t)$ and $i_6(t)$ capture the currents flowing through the electrical components C_1 to C_6, respectively. Next, we apply KVL on the three circuits L_1, L_2 and L_3 of the Δ-Y network topology (Figure 3(c)) to obtain the following equations:

$$\begin{aligned} &\text{Circuit L1} : v_1(t) + v_2(t) = -v_5(t) \\ &\text{Circuit L2} : v_4(t) + v_5(t) = v_6(t) \\ &\text{Circuit L3} : v_3(t) + v_4(t) = v_2(t) \end{aligned} \tag{2}$$

where $v_1(t)$ to $v_6(t)$ capture the voltages across the electrical components C_1 to C_6, respectively. Next, the matrix-based representation of the KCL using the incidence matrix for the Δ-Y network is given as follows:

$$\mathbf{A}\vec{I} = \vec{0} \tag{3}$$

where $\mathbf{A}$ represents the incidence matrix for the Δ-Y electrical network, $\vec{I}$ models a vector containing the currents flowing through all components of the network and $\vec{0}$ captures a vector with all of its entries as zeros. Similarly, the matrix-based representation of the KVL using the circuit matrix for the Δ-Y network is given as:

$$\mathbf{B}\vec{V} = \vec{0} \tag{4}$$

where $\mathbf{B}$ models the circuit matrix for the Δ-Y network topology and $\vec{V}$ provides a vector containing the voltages across all components of the network. Now, in order to formally analyze the Δ-Y network topology, we start with the formalization of a general representation of KCL in Isabelle/HOL as follows:

```
definition KCL:: (real ⇒ real) list ⇒ real ⇒ bool
where KCL Is t ≡ (Σi∈{0..<length Is}. (Is!i) t = 0)
```

where the function `KCL` accepts a list of currents `Is` across the components of a network and a time variable `t`, and returns a predicate providing the KCL, which states that the algebraic sum of all currents entering and leaving a node is equal to zero. Now, the KCL implementation of the Δ-Y network (Figure 3(c)) can be formally described as:

```
definition KCL_implem_delta_y where
  KCL_implem_delta_y Is t ≡ KCL [Is!1, Is!2, -(Is!0)] t ∧
                            KCL [Is!4, -(Is!1), -(Is!3)] t ∧
                            KCL [Is!3, Is!5, -(Is!2)] t ∧
                            KCL [Is!0, -(Is!4), -(Is!5)] t
```

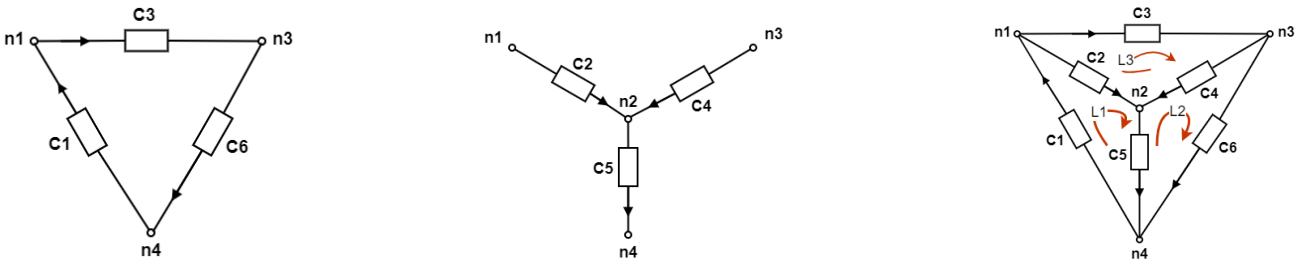
where `KCL_implem_delta_y` accepts the list of currents and applies KCL on nodes $n1$, $n2$, $n3$ and $n4$ of the Δ-Y network. Moreover, the list index in Isabelle/HOL starts from 0. Therefore, the variables $Is!0$ to $Is!5$ represent the currents passing through the components C_1 to C_6 of the electrical circuit network, respectively. Next, we formally verify the equivalence between the KCL implementation of the network and its mathematical representation (Equation (1)) as the following lemma in Isabelle/HOL:

```
lemma kcl_netw_eqs_delta_y:
KCL_implem_delta_y Is t ⟷ (Is!1) t + (Is!2) t = (Is!0) t ∧
(Is!1) t + (Is!3) t  = (Is!4) t ∧ (Is!3) t
+ (Is!5) t = (Is!2) t ∧ (Is!4) t + (Is!5) t = (Is!0) t
```

We now formalize the incidence matrix for the Δ-Y network (Figure 3(c)), which is further used to construct the matrix-based representation of the implementation of KCL. We verify the incidence matrix of the Δ-Y network as:

```
lemma incidence_matrix_delta_y:
 assumes nodes: 𝒩s = [n1,n2,n3,n4]
 and edges: ℰs = [(n4,n1),(n1,n2),(n1,n3),
                  (n3,n2),(n2,n4),(n3,n4)]
 shows incidence_matrix 𝒩s ℰs = mat_of_rows_list 6
                [[-1,1,1,0,0,0],[0,-1,0,-1,1,0],
                 [0,0,-1,1,0,1],[1,0,0,0,-1,-1]]
```

where the function `mat_of_rows_list` accepts an integer n, capturing the number of columns, and a list of list with length m, representing the number of rows, and provides a matrix of order $m \times n$. The verification of the above lemma is based on the definition of the incidence matrix, properties of the positive and negative incidences of the network system, the introduction rule `eq_matI`, and properties of the indices of the matrices alongside reasoning on lists and sets. Next, we



(a) (b) (c)

Fig. 3: (a) Δ Network Topology; (b) Y Network Topology; (c) Δ-Y Topology

formalize the matrix-based representation of the Kirchhoff's Laws (KCL and KVL) using the network topology (incidence and circuit) matrices as follows:

definition net_top_mat_implem **where**
net_top_mat_implem A X t ≡
(A *$_v$ (vecc X t)) = 0$_v$ (dim_row A)

where A is a real matrix (in our case, it is an incidence matrix for a KCL implementation and a circuit matrix for a KVL implementation), 0$_v$ (dim_row A) is a zero vector with a size equal to the row dimension of the matrix A, and the function vecc accepts the list X (list of currents and voltages for KCL and KVL implementations, respectively) and the real variable t and returns the vector of each list element at t. Now, we formally verify the equivalence of the mathematical representation of KCL (KCL applied on each node) and its matrix-based representation based on the incidence matrix in Isabelle/HOL as:

theorem KCL_implem_eq_inci_mat_rep_delta_y:
assumes nodes: $\mathcal{N}$s = [n1,n2,n3,n4]
and edges: $\mathcal{E}$s = [(n4,n1),(n1,n2),(n1,n3),
(n3,n2),(n2,n4),(n3,n4)]
shows net_top_mat_implem (incidence_matrix $\mathcal{N}$s $\mathcal{E}$s)
[Is!0,Is!1,Is!2,Is!3,Is!4,Is!5] t ⟷
KCL_implem_delta_y [Is!0,Is!1,Is!2,Is!3,Is!4,Is!5] t

The verification of the above theorem starts by splitting the equality in the goal into two implications. The first implication (⟶) is verified by performing row manipulation on the matrix representation of KCL to obtain the set of equations. The second implication (⟵) is simply verified by showing that the set of equations from KCL_imp_delta_y is equal to the rows of the incidence matrix. Similar to the KCL, we formalize the generalized representation of KVL in Isabelle/HOL to ensure that the algebraic sum of the voltages across any set of edges in a closed loop is zero.

definition KVL:: (real ⇒ real) list ⇒ real ⇒ bool
where KVL Vs t ≡ (Σi∈{0..<length Vs}. (Vs!i) t = 0)

Now, the KVL implementation of the Δ-Y network toplogy (Figure 3(c)) is formalized in Isabelle/HOL as:

definition KVL_implem_delta_y **where**
KVL_implem_delta_y Vs t ≡ KVL [Vs!0, Vs!1, Vs!4] t ∧
KVL [Vs!3, Vs!4, -(Vs!5)] t ∧ KVL [Vs!2, Vs!3, -(Vs!1)] t

where KVL_implem_delta_y accepts the list of voltages and applies KVL on the circuits $L1$, $L2$ and $L3$ of the Δ-Y network. Next, we formally verify the equivalence between the KVL implementation of the network topology and its mathematical representation (Equation (2)) as the following theorem in Isabelle/HOL.

lemma kvl_netw_eqs_delta_y:
KVL_implem_delta_y Vs t ⟷ (Vs!0) t + (Vs!1) t = -(Vs!4) t ∧
(Vs!3) t + (Vs!4) t = (Vs!5) t ∧ (Vs!2) t + (Vs!3) t = (Vs!1) t

Next, we formally verify the circuit matrix of the Δ-Y network depicted in Figure 3(c) and derive the matrix-based representation of Kirchhoff's Voltage Law (KVL) using this circuit matrix. Finally, the equality between the equation-based representation of KVL and its circuit matrix-based representation is verified as the following theorem in Isabelle/HOL.

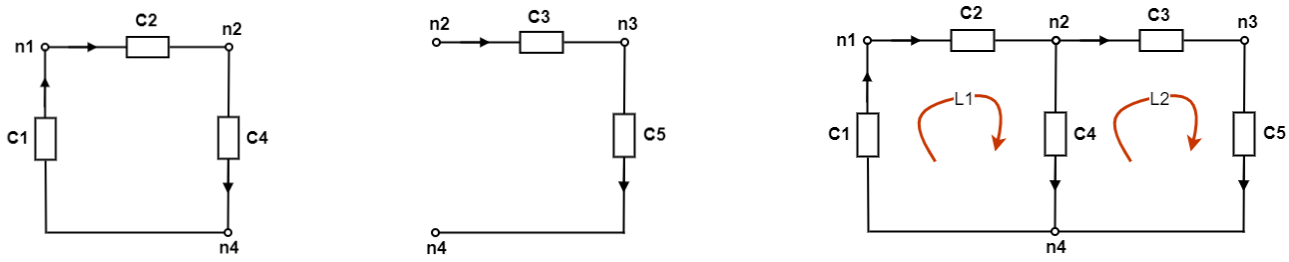
theorem KVL_implem_eq_circuit_mat_rep_delta_y:
assumes circuits: $\mathcal{L}$s = [[(n4,n1),(n1,n2),(n2,n4)],
[(n3,n4),(n4,n2),(n2,n3)],
[(n1,n3),(n3,n2),(n2,n1)]]
and edges: $\mathcal{E}$s = [(n4,n1),(n1,n2),(n1,n3),
(n3,n2),(n2,n4),(n3,n4)]
shows net_top_mat_implem (circuit_matrix $\mathcal{L}$s $\mathcal{E}$s)
[Vs!0,Vs!1,Vs!2,Vs!3,Vs!4,Vs!5] ⟷
KVL_implem_delta_y [Vs!0,Vs!1,Vs!2,Vs!3,Vs!4,Vs!5] t

The proof process of the above theorem is very similar to that of one verifying an equivalence relationship between KCL equations and the network topology-based matrix representation of the Δ-Y network.

B. Formal Analysis of the Π-L Network Topology

Figure 4(c) depicts a Π-L network topology, which consists of five components $C1$, $C2$, $C3$, $C4$ and $C5$, shown as rectangular boxes. It is constructed by a series combination of the Π (Figure 4(a)) and L (Figure 4(b)) topologies. Moreover, a judicious selection of these components results into different electrical networks required for performing different tasks. For example, for a choice of $C1$ as a voltage/current source, $C2$ as a resistor and $C4$ as a capacitor, the corresponding Π topology results into an RC (resistor-capacitor) circuit or RC filter, which is widely used to process the input signal by allowing passage of only a certain range of frequencies.

We start the formal analysis of the Π-L network by formalizing its KCL implementation in Isabelle/HOL as:



(a) (b) (c)

Fig. 4: (a) Π Network Topology; (b) L Network Topology; (c) Π-L Topology

```
definition KCL_implem_pi_l where
  KCL_implem_pi_l Is t ≡ KCL [Is!1, -(Is!0)] t ∧
 KCL [Is!2, -(Is!1), Is!3] t ∧ KCL [Is!4, -(Is!2)] t ∧
 KCL [Is!0, -(Is!3), -(Is!4)] t
```

where KCL_implem_pi_l accepts a list of currents and applies KCL on nodes $n1$, $n2$, $n3$ and $n4$ of the Π-L network. Next, we formally verify the equivalence between the KCL implementation of the network and its mathematical representation as the following lemma:

```
lemma kcl_netw_eqs_pi_l: KCL_implem_pi_l Is t ⟷
  (Is!1) t = (Is!0) t ∧ (Is!2) t + (Is!3) t = (Is!1) t ∧
  (Is!2) t = (Is!4) t ∧ (Is!3) t + (Is!4) t = (Is!0) t
```

Next, we formalize the incidence matrix for the Π-L network (Figure 4(c)). Moreover, we formally verify the matrix-based representation of the KCL using this incidence matrix. Then, we formally verify the equivalence of the mathematical representation of KCL (KCL applied on each node) and its matrix-based representation based on the incidence matrix as the following lemma in Isabelle/HOL:

```
theorem KCL_implem_eq_inci_mat_rep_pi_l:
assumes nodes: 𝒩s = [n1,n2,n3,n4]
and edges: ℰs = [(n4,n1),(n1,n2),(n2,n3),
                                (n2,n4),(n3,n4)]
shows net_top_mat_implem (incidence_matrix 𝒩s ℰs)
               [Is!0,Is!1,Is!2,Is!3,Is!4] t ⟷
      KCL_implem_pi_l [Is!0,Is!1,Is!2,Is!3,Is!4] t
```

The verification of the above theorem is very similar to that of theorem KCL_implem_eq_inci_mat_rep_delta_y verified for the Δ-Y network topology. Next, the KVL implementation of the Π-L network topology (Figure 4(c)) is formalized in Isabelle/HOL as:

```
definition KVL_implem_pi_l where
  KVL_implem_pi_l Vs t ≡ KVL [Vs!0, Vs!1, Vs!3] t ∧
                         KVL [Vs!2, Vs!4, -(Vs!3)] t
```

where KVL_implem_pi_l accepts the list of voltages and applies KVL on the circuits $L1$ and $L2$ of the Π-L network. Next, we formally verify the equivalence between the KVL implementation of the network topology and its mathematical representation as the following lemma in Isabelle/HOL.

```
lemma kvl_netw_eqs_pi_l:
  KVL_implem_pi_l Vs t ⟷ (Vs!0) t + (Vs!1) t = -(Vs!3) t ∧
                         (Vs!2) t + (Vs!4) t = (Vs!3) t
```

Based on the two circuits depicted in the Π-L network (Figure 4(c)), we now formalize the corresponding circuit matrix and formally verify the matrix-based representation of the KVL. Thereafter, the equivalence between the equation-based and circuit matrix-based representations of KVL is formally verified as the following theorem in Isabelle/HOL.

```
theorem KVL_implem_eq_circuit_mat_rep_pi_l:
assumes networks: ℒs = [[(n4,n1),(n1,n2),(n2,n4)],
                        [(n2,n3),(n3,n4),(n4,n2)]]
and edges: ℰs = [(n4,n1),(n1,n2),(n2,n3),
                                (n2,n4),(n3,n4)]
shows net_top_mat_implem (circuit_matrix ℒs ℰs)
                [Vs!0,Vs!1,Vs!2,Vs!3,Vs!4] ⟷
      KVL_implem_pi_l [Vs!0,Vs!1,Vs!2,Vs!3,Vs!4] t
```

The proof process of the above theorem is very similar to that of one verifying an equivalence relationship between KVL equations and the network topology-based matrix representation of the Δ-Y network topology. To the best of our knowledge, this is the first formal analysis of electrical circuit network topologies based on the topological matrices using HOL theorem proving. The Isabelle/HOL code for formalization and verification efforts presented in this paper are available at [20].

VI. Discussion

Network topology matrices are an efficient tool for analyzing electrical circuit networks. Moreover, considering the growing range of graph applications using matrices in many areas, the formalization of mathematical foundations establishing a relationship between graphs and matrices is desired. Therefore, we aim to build a formal library that integrates between graph theory, matrix theory and electrical circuit network theory. We have been able to successfully formalize the widely used incidence and circuit matrices, which are further used for analyzing various network topologies, as illustrated in Section V of the paper. Another advantage of our proposed formalization is the generic nature of the formal analysis of electrical circuit networks. For example, the formally verified properties, i.e., all lemmas regarding the dimensions and indexing are verified for universally quantified variables and functions that could further be specialized to obtain results for a particular scenario. However, in the case

of traditional computer-simulation based analysis, we need to model each circuit individually.

Similarly, all lemmas/theorems regarding the formal analysis of various network topologies are verified for generic electrical components that could be replaced with any electrical components, such as resistors, capacitors, inductors and current/voltage sources in order to obtain results for a particular circuit. For example, in Figure 3(c), by setting $C1$ as a current/voltage source, $C3$ and $C4$ as resistors, $C5$ and $C6$ as inductors and $C2$ as a capacitor, the corresponding Δ-Y network topology results into a widely used RLC (resistor, inductor, capacitor) circuit network depicted in Figure 5. For the analysis of this RLC circuit network, we only need to specialize the current and voltage functions in the relevant theorems regarding KCL and KVL implementations, presented in Section V, with the currents and voltages across these components, i.e., resistors, inductors and capacitors.

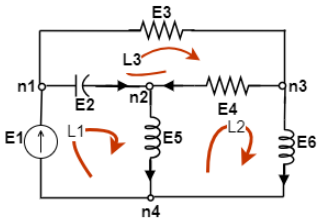


Fig. 5: An RLC Network

A further advantage of our proposed approach is the assurance of explicit presence of all required assumptions in the definitions of the network incidence and circuit systems, alongside the related lemmas that may get overlooked in traditional paper-and-pencil based proof methods. One of the main challenges encountered during the proposed formalization is the need to explicitly express various mathematical concepts, such as the circuit formulation (cf. Sections IV) that are presented in the literature somehow abstractly, in order to make them amenable to the formalization in Isabelle/HOL.

VII. Conclusion

In this paper, we proposed to use higher-order-logic (HOL) theorem proving for formally analyzing electrical circuit networks using network topology matrices. Firstly, we developed the network incidence and the circuit systems with their constraints using the locale modules of Isabelle/HOL. We also verified related system properties within the locale. Next, we formalized the respective system's matrices of networks, represented by directed graphs. Next, we used the formalization of the topological matrices for formally analyzing various electrical network topologies, such as the Δ-Y and Π-L topologies. In future, we plan to formally analyze more network topologies, such as H, Box and Bridge topologies [2] to name a few, with an aim to build a comprehensive library for electrical circuit networks analysis. Another direction can be to extend our network topology system by formalizing other topological matrices, like cutset and path matrices [3] and verifying more properties on top of these matrices to facilitate the analysis of safety-critical systems.

References

[1] L. J. Tung and B. W. Kwan, *Circuit Analysis*. World Scientific, 2001.
[2] C. J. Monier, *Electric Circuit Analysis*. Pearson, 2001.
[3] W.-K. Chen, *Graph Theory and its Engineering Applications*. World Scientific, 1997, vol. 5.
[4] F. E. Hohn, *Elementary Matrix Algebra*. Courier Corporation, 2013.
[5] T. Nipkow, M. Wenzel, and L. C. Paulson, *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Springer, 2002.
[6] K. H. Rosen, *Discrete Mathematics & Applications*. McGraw-Hill, 1999.
[7] M. H. Zaki, O. Hasan, S. Tahar, and G. Al-Sammane, "Framework for Formally Verifying Analog and Mixed-signal Designs," *Computational Intelligence in Analog and Mixed-Signal (AMS) and Radio-Frequency (RF) Circuit Design*, pp. 115–145, 2015.
[8] S. H. Taqdees and O. Hasan, "Formally Verifying Transfer Functions of Linear Analog Circuits," *IEEE Design & Test*, vol. 34, no. 5, pp. 30–37, 2017.
[9] A. Rashid and O. Hasan, "Formalization of Lerch's Theorem using HOL Light," *Journal of Applied Logics*, vol. 5, no. 8, p. 1623, 2018.
[10] A. Rashid, A. Gauhar, and O. Hasan, "FASiM: A Framework for Automatic Formal Analysis of Simulink Models of Linear Analog Circuits," in *International Systems Conference*. IEEE, 2020, pp. 1–7.
[11] L. Noschinski, "A Graph Library for Isabelle," *Mathematics in Computer Science*, vol. 9, no. 1, pp. 23–39, 2015.
[12] P. Lammich and S. R. Sefidgar, "Formalizing Network Flow Algorithms: A Refinement Approach in Isabelle/HOL," *Journal of Automated Reasoning*, vol. 62, pp. 261–280, 2019.
[13] K. H. Rosen, *Handbook of Discrete and Combinatorial Mathematics*. CRC Press, 1999.
[14] J. Heras, M. Poza, M. Dénès, and L. Rideau, "Incidence Simplicial Matrices Formalized in Coq/SSReflect," in *Intelligent Computer Mathematics: LNCS*, vol. 6824. Springer, 2011, pp. 30–44.
[15] C. Edmonds and L. C. Paulson, "Formalising Fisher's Inequality: Formal Linear Algebraic Proof Techniques in Combinatorics," in *Interactive Theorem Proving, LIPIcs*, vol. 237, 2022, pp. 11:1–11:19.
[16] L. Babai and P. Frankl, *Linear Algebra Methods in Combinatorics*. Department of Computer Science, The University of Chicago, 2022.
[17] S. Seshu and M. B. Reed, *Linear Graphs and Electrical Networks*. Addison-Wesley Publishing Company, 1961.
[18] C. Ballarin, "Locales: A Module System for Mathematical Theories," *Journal of Automated Reasoning*, vol. 52, no. 2, pp. 123–153, 2014.
[19] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*. Courier Dover Publications, 2017.
[20] K. Aksoy, "Isabelle/HOL Proof Script: Formal Analysis of Electrical Circuit Network Topologies using Theorem Proving," https://hvg.ece.concordia.ca/code/Isabelle-hol/ecnt.zip, 2025.
[21] P. S. Farago, *An Introduction to Linear Network Analysis*. English Universities Press, 1961.