Linear Network Reduction Via Generalized Y-Δ Transformation: Theory

Zhanhai Qin
zqin@cs.ucsd.edu

Chung-Kuan Cheng
kuan@cs.ucsd.edu

Department of Computer Science & Engineering
University of California, San Diego
9500 Gilman Drive, La Jolla, CA 92093

May 2002

Abstract

This report provides the theoretical foundation to STAR program, a linear network reduction software kit. Y-Δ transformation in s-domain is the essential part of the reduction engine of the program. We have extended it to handle current/voltage sources and K elements[22]. Coming with it are node ordering algorithm and some novel numerical stability control techniques. They play an important role in the overall reduction flow by providing the consistent accuracy in program outputs. [1] is dedicated to the applications to the program.

Keyword: Wye-Delta Transformation, Interconnect Model Reduction, Symbolic Network Analysis.

Contents

1 Introduction **3**

2 Y-Δ Transformation **4**
2.1 Example 4
2.2 Branch with Current/Voltage Source 5
2.3 Branch with K element 6
2.4 Generalization 7

3 Node Ordering **8**
3.1 MMD Algorithm 8

4 Common Factor in Y-Δ Transformation **9**
4.1 Example 10
4.2 ω Exists in General Graphs 13
4.3 Common Factors in Denominators Only 18

5 Overall Algorithm **18**

6 Admittance in Its Simplest Form **20**

7 Conclusion **23**

1 Introduction

Due to the increasing complexity of VLSI chips, a linear network such as a power/ground grid usually contains millions of RLC elements generated from extraction tools. Lower supply voltages make the voltage variation across the power grids very critical because large voltage drop reduces the supply voltage at some logic gates, leading lower noise margins and resulting in a serious performance impact[2, 3]. Stronger coupling effects across deep sub-micron interconnects demand accurate and efficient simulation as well. On the other hand, SPICE[4] as the standard of circuit-level simulation tools, takes hours and consumes gigabytes of memory on a modern workstation to simulate a moderate size RLC network[5]. Simulating such circuits becomes a challenging task.

To work around the poor performance of SPICE, two strategies are commonly used: (1) to increase the efficiency of solving the MNA(modified nodal analysis) or NA(nodal analysis) formulated system equations; (2) to reduce the size of original networks via model reduction techniques.

MNA using LU factorization in SPICE was shown less efficient than NA using preconditioned Krylov-subspace iterative methods[6]. NA using SuperLU[7] factorization in [5] provided comparable performance to iterative methods while the robustness of direct methods was kept. [8, 9] explored the regular grid structure of power and ground networks and used multigrid technique to solve a coarse grid and map the solution back to the original fine grid. These approaches fall into the first category.

The moment-matching technique, on the other hand, has been widely used to approximate waveforms of a linear interconnect network using its lower order moments[10, 11, 12]. Since the advent of the technique, many interconnect delay evaluation models[13, 14, 15] were proposed. It was well known that the moment-matching technique was equivalent to a Padé approximation, which may generate positive poles for an originally passive circuit. [16] partitioned RC interconnect networks and reduced each sub-network into a macromodel, reserving lower orders of the port admittance matrix Y. The method guaranteed the realizability of the macromodels for RC circuits.

MPVL (matrix Padé via Lanczos)[17], block Arnoldi[18] and PRIMA[19] are admittance matrix ($Y(s)$) based model reduction methods so that they perform model order reduction on each entry in $Y(s)$ simultaneously. The PACT algorithm[20] first introduced congruence transformations for order reduction of RC circuits. The same authors proposed split congruence transformations[21] for passive reductions of RLC circuits.

We have proposed a new RKC network reduction method. The principal idea is that we consider a linear network as a graph and perform Y-Δ transformation on each node of no interest until all such kind of nodes are eliminated. The generalized Y-Δ transformation formula is able to handle current/voltage sources and K elements[22]. Nodes eligible to be eliminated are called *internal* nodes. and others are called *external* nodes. Different from topological formulas for network functions [27], our approach keeps only low-order coefficients of them. Fortunately, these coefficients are exactly the same as they were computed without discarding any high-order terms. A complexity comparison of the two approaches is given in Section 5 after we give the proposed algorithm.

Generally speaking, the input admittance of an one-port N-th order RLC linear time invariant network in s-domain is a N-th order rational function $Y(s)$. Y-Δ transformation reduces the network in term of the number of nodes, a straightforward implementation of the approach, however, leads to a rational function $Y'(s)$ whose order is far beyond N (about $N!$). It is worthy noting that $Y(s) = Y'(S)$ indeed. By exploiting the structure of Y-Δ transformation process, we have found out that a lot of common factors are introduced into the numerator and denominator of $Y'(s)$, which actually should be canceled out. This finding and some other practical numerical considerations allow us to control round-off errors in high-order polynomial computation. It is also crucial in pole/residue analysis, as shown in [1].

Our main contributions are:

- admittance is always kept in its rational form and all the coefficients are the same as they were computed using exact symbolic approaches without discarding any high-order terms;
- With impedance realization method[1], incremental simulation can be achieved by simply specifying nodes in tunable sub-circuits as external nodes;
- Pole/Residue sensitivity analysis can be achieved inexpensively along the reduction process.

The remaining of this report is organized as follows. In the next section, we briefly review the fundamentals of Y-Δ transformation and the generalized formula. Multiple minimum degree (MMD) algorithm [25] is covered in the Section 4. And Section 5 is dedicated to explaining the existence of common factors in transformed admittance. The overall reduction algorithm will be given in Section 6. Section 7 concludes the proposed work.

2 Y-Δ Transformation

2.1 Example

Before we present the general form of Y-Δ transformation, we first illustrate a simple numerical example.

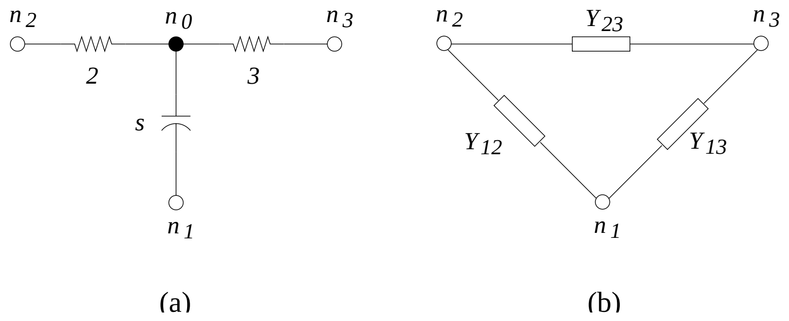


Figure 1: A numerical example for Y-Δ transformation: (a)circuit schematic before the transformation; (b)circuit schematic after the transformation.

As shown in Fig. 1(a), n_0 is adjacent to n_1, n_2, and n_3 *only*. To simplify our explanation, we further assume that there is no admittance between any two of n_1, n_2 and n_3. KCL (Kirchhoff's Current Law) equations for node n_0, n_1, n_2, and n_3 can be established as follows:

$$(5+s)V_0 - sV_1 - 2V_2 - 3V_3 = 0 \tag{1}$$
$$-sV_0 + (s+Y_1)V_1 - \boldsymbol{Y}_1\boldsymbol{V}_x = I_1 \tag{2}$$
$$-2V_0 + (2+Y_2)V_2 - \boldsymbol{Y}_2\boldsymbol{V}_x = I_2 \tag{3}$$
$$-3V_0 + (3+Y_3)V_3 - \boldsymbol{Y}_3\boldsymbol{V}_x = I_3 \tag{4}$$

In the equations, we have denoted $\boldsymbol{V}_x$ as node voltage of $n_4, \ldots n_n$. $\boldsymbol{Y}_1$, $\boldsymbol{Y}_2$ and $\boldsymbol{Y}_3$ are admittance vectors with the i-th entry equal to the admittance between n_1, n_2, n_3 and n_i, $i \geq 4$, respectively. Y_1, Y_2 and Y_3 are the total admittance between n_1, n_2, n_3 and nodes except n_0, respectively. Finally, I_1, I_2 and I_3 are the total current injecting into n_1, n_2 and n_3 from current sources, respectively. From 1, we can denote V_0 in terms of V_1, V_2, and V_3 as:

$$V_0 = \frac{sV_1 + 2V_2 + 3V_3}{5+s}. \tag{5}$$

Inserting (5) into (2)–(4) gives

$$(\frac{5s}{5+s} + Y_1)V_1 - \frac{2s}{5+s}V_2 - \frac{3s}{5+s}V_3 - \boldsymbol{Y}1\boldsymbol{V}_x = I_1 \tag{6}$$
$$-\frac{2s}{5+s}V_1 + (\frac{6+2s}{5+s} + Y2)V_2 - \frac{6}{5+s}V_3 - \boldsymbol{Y}2\boldsymbol{V}_x = I_2 \tag{7}$$
$$-\frac{3s}{5+s}V_1 - \frac{6}{5+s}V_2 + (\frac{6+3s}{5+s} + Y3)V_3 - \boldsymbol{Y}3\boldsymbol{V}_x = I_3 \tag{8}$$

Comparing (6)–(8) with Fig. 1(b), we can find out that

$$\begin{cases} Y_{12} &= 2s/(5+s) \\ Y_{13} &= 3s/(5+s) \\ Y_{23} &= 6/(5+s). \end{cases}$$

From the example, you may have noticed that what we performed is equivalent to one column/row Gauss elimination to the system equations. When actual Y-Δ transformations are carried out, we do not formulate a circuit into simultaneous system equations. Instead we consider it as a graph and operate on the graph directly.

One advantage of our approach over SPICE is that once massive internal nodes are eliminated while external nodes are preserved, then after impedance realization is applied to transformed impedance, the resultant passive circuit would be much smaller and has similar characteristics in the working frequency range. And because any node can be specified as external node to be preserved, incremental simulation becomes easier. For example in a datapath structure with hundreds of bi-directional divers exist in the same interconnect network. We can collapse the internal structure of the network. Then delays after tuning physical parameters of drivers can be simulated very easily using SPICE as the overall circuit is much smaller than the original one.

Simple circuit elements, i.e. resistors, capacitors, and self partial inductors, have well-known admittance forms in s-domain. But Y-Δ transformations involving current/voltage sources and mutual partial inductors are not straightforward. Particularly, even though the generalized Y-Δ transformation is able to handle mutual inductors, including them prevents us from giving a simple and unified transformation formula.

K-based inductance extraction method[22] proposed a new circuit element — K element to capture the inductance effects of interconnects in integrated circuits. (14) of [23] gives the branch equation for element K:

$$Kv = \frac{di}{dt}, \tag{9}$$

which in s-domain can be written as:

$$\frac{K}{s}V = I. \tag{10}$$

Although V and I refer to different branches for capturing mutual inductance effects, a simple conversion will integrate K elements into our transformation formula seamlessly.

2.2 Branch with Current/Voltage Source

Branches involved in Y-Δ transformation can not only include resistors, capacitors and self inductors, but also current/voltage sources and mutual inductors. We will give the Y-Δ transformation formula for circuits with current and voltage sources in this sub-section, and K elements in the next sub-section.

Following the similar procedure in Section 3, we apply Y-Δ transformation to node n_0 in Fig. 2(a),

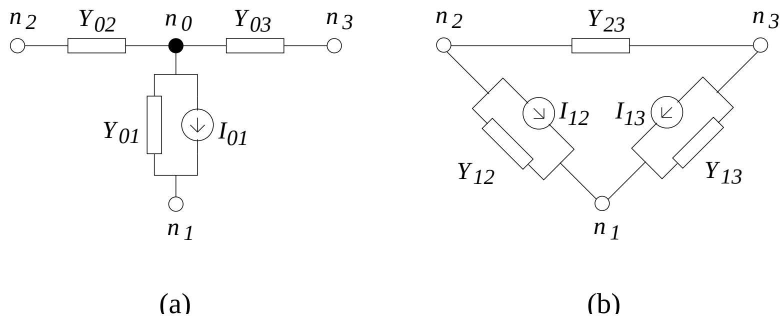


Figure 2: Y-Δ transformation with current source involved: (a)circuit schematic before the transformation; (b)circuit schematic after the transformation.

$$\begin{cases} Y_{12} &= Y_{01}Y_{02}/(Y_{01}+Y_{02}+Y_{03}) \\ Y_{13} &= Y_{01}Y_{03}/(Y_{01}+Y_{02}+Y_{03}) \\ Y_{23} &= Y_{02}Y_{03}/(Y_{01}+Y_{02}+Y_{03}) \\ I_{12} &= Y_{02}/(Y_{01}+Y_{02}+Y_{03})I_{01} \\ I_{13} &= Y_{03}/(Y_{01}+Y_{02}+Y_{03})I_{01}. \end{cases} \tag{11}$$

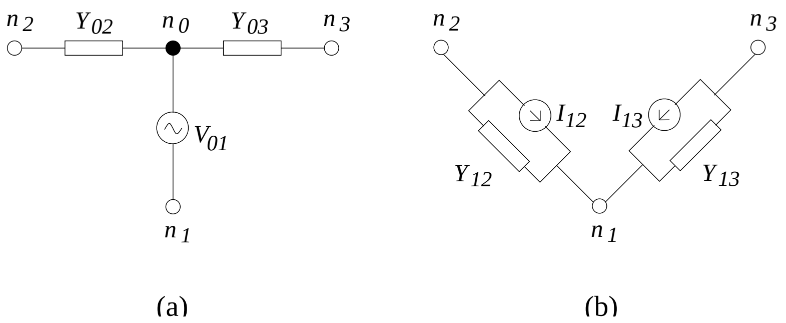


Figure 3: Y-Δ transformation with voltage source involved: (a)circuit schematic before the transformation; (b)circuit schematic after the transformation.

And performing Y-Δ transformation to node n_0 in Fig. 3(a) gives

$$\begin{cases} Y_{12} &= Y_{02} \\ Y_{13} &= Y_{03} \\ I_{12} &= Y_{02}V_{01} \\ I_{13} &= Y_{03}V_{01}. \end{cases} \tag{12}$$

One can also derive (11) and (12) from Norton's theorem. A generalization of the two transformation formulas will be given in Theorem 1.

2.3 Branch with K element

Self K elements are considered with no difference from others such as resistors and capacitors in Y-Δ transformation. But for mutual K, we have to do a conversion on it. For the example shown in Fig. 4(a), the circuit

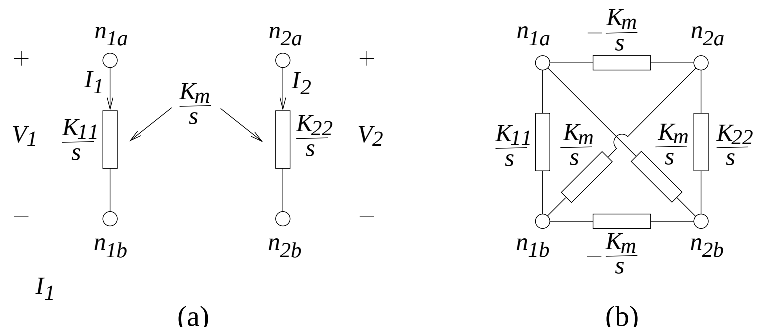


Figure 4: Conversion on mutual K in s-domain: (a)given mutual K; (b) converted K.

branch equations are written as

$$\begin{cases} \frac{K_{11}}{s}V_1 + \frac{K_m}{s}V_2 = I_1 \\ \frac{K_m}{s}V_1 + \frac{K_{22}}{s}V_2 = I_2, \end{cases} \tag{13}$$

which can be rewritten as KCL equations for the four nodes in Fig. 4(a) as

$$\begin{cases} \frac{K_{11}}{s}V_{1a} - \frac{K_{11}}{s}V_{1b} + \frac{K_m}{s}V_{2a} - \frac{K_m}{s}V_{2b} = I_1 \\ -\frac{K_{11}}{s}V_{1a} + \frac{K_{11}}{s}V_{1b} - \frac{K_m}{s}V_{2a} + \frac{K_m}{s}V_{2b} = -I_1 \\ \frac{K_m}{s}V_{1a} - \frac{K_m}{s}V_{1b} + \frac{K_{11}}{s}V_{2a} - \frac{K_{11}}{s}V_{2b} = I_2 \\ -\frac{K_m}{s}V_{1a} + \frac{K_m}{s}V_{1b} - \frac{K_{11}}{s}V_{2a} + \frac{K_{11}}{s}V_{2b} = -I_2. \end{cases} \tag{14}$$

One can find out that the KCL equations for the four nodes in Fig. 4(b) are exactly the same as (14, so that the circuit in Fig. 4(b) is equivalent to the circuit in Fig. 4(a). Although some values in (b) have negative signs,the equivalent circuits are still passive because K-based method guarantees the extracted K matrix to be positive definite, the equivalent.

2.4 Generalization

Now we state a generalized Y-Δ transformation formula including linear(ized) current/voltage sources, resistors, capacitors, and K elements. Although the theorem is stated based on the simple scenario with a single current/voltage source, for complicated cases, one can always use them in conjunction with superposition theorem.

THEOREM 1. *With no loss of generality, let n_0 be the node that we want to eliminate, let $n_1, n_2, \ldots, n_k$ be the adjacent nodes to n_0. Y_{ij} denotes the admittance between node n_i and n_j. Thus Y_{01}, Y_{02}, ..., Y_{0k} are the admittance between $n_1, n_2, \ldots, n_k$ and n_0, respectively. Particularly, a current source is considered to be open-circuited and a voltage source short-circuited in terms of admittance. In s-domain, admittance is a function of s.*

After n_0 is eliminated, $n_1, n_2, \ldots, n_k$ become pairwise adjacent and form a clique. A set of admittance

$$\{Y_{ij} \mid i, j \in [1,k], i < j\}$$

are generated, and

$$Y_{ij}(s) = Y_{ij}(s) + \frac{Y_{0i}(s) \times Y_{0j}(s)}{Y_{01}(s) + Y_{02}(s) + \cdots + Y_{0k}(s)}. \tag{15}$$

Suppose I_{01} was a current source between n_0 and n_1 before the elimination. Then after it, a set of current sources

$$\{I_{1j} \mid j \in [2,k]\}$$

have to be generated, and

$$I_{1j}(s) = \frac{Y_{0j}(s)}{Y_{01}(s) + Y_{02}(s) + \cdots + Y_{0k}(s)} I_{01}(s). \tag{16}$$

Alternatively, suppose V_{01} was a voltage source between n_0 and n_1 before the elimination. Then after it, a set of current sources

$$\{I_{1j} \mid j \in [2,k]\}$$

have to be generated, and

$$I_{1j}(s) = Y_{0j}(s) V_{01}(s). \tag{17}$$

■

A useful observation from Theorem 1 is that different from Padé approximation, using Y-Δ transformation, coefficients of admittance are derived directly from admittance in original circuits and are kept in its original rational form. By matching the lower-order coefficients, the method can capture complex poles of original circuits near imaginary axle accurately.

COROLLARY 1. *If all RLC elements in a given linear RLC system are of positive values, no matter how many nodes are eliminated via Y-Δ transformation, the transformed admittance between any two nodes n_i and n_j can be written as*

$$\frac{a_0 + a_1 s + \cdots + a_m s^m}{b_0 + b_1 s + \cdots + b_n s^n}, \tag{18}$$

where $b_0 > 0$, and $a_i, b_j \geq 0, i \in [0,m]$ and $j \in [1,n]$. ■

The above corollary holds immediately after the given theorems.

3 Node Ordering

As elaborated after the example in Section 3, eliminating nodes in an interconnect network via Y-Δ transformation is equivalent to LU factorizing the corresponding MNA formulated system equations. Non-zero fill-ins in LU factorization correspond to new branches among nodes in the network. Hence given a linear network, the order in which nodes are eliminated is very important in that different orders generally lead to different numbers of new branches. And the complexity of every Y-Δ transformation on a node n_i is $O(|n_i|^2)$, where $|n_i|$ is the current degree of n_i. Because we do not perform Y-Δ transformation on every single node in a system as external ones have to be preserved, We revised MMD algorithm to fit our needs.

3.1 MMD Algorithm

The most widely used general-purpose ordering scheme is the minimum-degree algorithm [24]. It is a heuristic algorithm, but it is very successful in reducing non-zero fill-ins in LU factorization. The scheme attempts to reduce the fill-ins of a given matrix by a local minimization of non-zeros in the factored matrix. It is used as a practical approximate solution to the NP-complete *fill-in minimization problem* [26].

The concept of indistinguishable nodes [24] is developed to eliminate a subset of nodes all at the same time (Step 3) instead of just one node of the minimum degree. In the elimination process, nodes n_i and n_j that satisfy

$$Adj(n_i) \cup \{n_i\} = Adj(n_j) \cup \{n_j\}$$

in a graph are said to become indistinguishable. These nodes can be numbered consecutively in the minimum-degree ordering.

```
Step 1. (Initialization) Initialize the set of eliminated nodes $S = \emptyset$, and the set of uneliminated nodes $X$ includes
        all internal nodes.
        Compute the degree of all the nodes in $X$.

Step 2. (Minimum Degree) Determine the new minimum degree among nodes in $X$ and the set $T$ of all nodes
        in the set $X - S$ of the minimum degree.

Step 3. (Mass Elimination) All nodes in $X$ are unflagged.
        For each node $n_i$ in $T$:
          If node $n_i$ is unflagged
            find the set $W$ of indistinguishable nodes of $n_i$;
            flag the adjacent nodes of $n_i$ and the nodes of in the current graph;
            $S = S \cup W$.

Step 4. (Degree update) Determine the representation of the new graph.
        Update the degree of all the flagged nodes in $X - S$ that have not been outmatched.

Step 5. (Loop or Stop) Repeat steps 2 to 4 until $T$ is empty.
```

Note that we excluded external nodes from the the set of uneliminated nodes. So that the resultant node elimination sequence contains internal nodes only.

Theorem 2. *Let S_1 and S_2 denote any different node elimination sequences of a given circuit. Suppose n_i and n_j are two external nodes of the circuit, Let Y_{ij} and Y'_{ij} are the admittance between n_i and n_j after Y-Δ Transformations following sequence S_1 and S_2, respectively. The following equation holds:*

$$Y_{ij} = Y'_{ij}.$$

■

The theorem tells us that even although different node elimination sequences could have dramatically different impact on the performance of reduction via Y-Δ Transformation, the transformed admittance from these different reduction sequences are the same.

An observation from (15) is that without considering common factor cancellation between Y_{ij}'s numerator and denominator, the order of Y_{ij} is the summation of the order of Y_{i0} and Y_{j0}. Because the reduction is to be applied to each internal nodes, Y_{ij} may be appearing on the right-hand side of (15) so that order of transformed admittance will be growing fast. When reducing interconnect networks, on the other hand, we only need to keep coefficients of Y_{ij}'s lower order terms, i.e., $\{a_0, a_1, \ldots, a_k\}$ and $\{b_0, b_1, \ldots, b_k\}$ of Y_{ij} in (18). Most interconnect reduction models have $k \leq 3$. The following theorem ensures us that no matter transformed admittance Y_{ij} is an intermediate admittance or a final one to be realized, keeping its lower k order coefficients in its numerator and denominator throughout the whole reduction process delivers correct lower k order coefficients of final transformed admittance.

Theorem 3. *With no loss of generality, let us refer to (15). Suppose we have two Y-Δ reduction procedures A and B. In A, a newly transformed admittance is termed as Y_{ij} and can be computed as*

$$Y_{ij}(s) = \frac{Y_{i0}(s) \times Y_{j0}(s)}{Y_{10}(s) + Y_{20}(s) + \cdots + Y_{k0}(s)}.$$

While in B, a newly transformed admittance is termed as $\tilde{Y}_{ij}$ and can be computed as

$$\tilde{Y}_{ij}(s) \approx Y'_{ij}(s) = \frac{\tilde{Y}_{i0}(s) \times \tilde{Y}_{j0}(s)}{\tilde{Y}_{10}(s) + \tilde{Y}_{20}(s) + \cdots + \tilde{Y}_{k0}(s)}.$$

Here Y'_{ij} is in the form

$$Y'_{ij}(s) = \frac{a_0 + a_1 s + \cdots + a_m s^m}{b_0 + b_1 s + \cdots + b_n s^n}.$$

And $\tilde{Y}_{ij}$ is the k-th order approximate of Y'_{ij}

$$\tilde{Y}_{ij}(s) = \frac{a_0 + a_1 s + \cdots + a_k s^k}{b_0 + b_1 s + \cdots + b_k s^k}, \qquad 0 \leq k \leq \min(m, n).$$

If $\tilde{Y}_{i0}$, $\tilde{Y}_{j0}$, $\tilde{Y}_{10}$, $\tilde{Y}_{20}, \ldots, \tilde{Y}_{k0}$ are the k-th order approximate of Y_{i0}, Y_{j0}, $Y_{10}, Y_{20}, \ldots, Y_{k0}$, respectively, then $\tilde{Y}_{ij}$ is also the k-th order approximate of Y_{ij}. ■

The theorem can be proven using mathematical induction.

4 Common Factor in Y-Δ Transformation

Briefed in the introduction, Y-Δ transformation process discussed so far introduces common factors into the numerator and denominator of the right-hand side admittance in (15). This side effect is harmful to our reduction algorithm because (1)they cause the magnitude of coefficients of the numerator and denominator unnecessarily grow: basically they increase exponentially along with the order of the corresponding terms; (2) common factors in numerators/denominators create fake zeros/poles that hamper the pole/residue analysis [1].

In this section, we treat linear networks as graphs, representing admittance of the i-th branch as a_i/b_i. Because the Y-Δ transformation is an continuous process, we denote admittance of original circuits as $Y^{(0)}_{i,j}$, and $Y^{(1)}_{i,j}$ when the first node is eliminated. In this way, we can rewrite (15)

$$Y^{(t)}_{i,j}(s) = Y^{(t-1)}_{i,j}(s) + \frac{Y^{(t-1)}_{t-1,i}(s) \times Y^{(t-1)}_{t-1,j}(s)}{Y^{(t-1)}_{t-1,1}(s) + Y^{(t-1)}_{t-1,2}(s) + \cdots + Y^{(t-1)}_{t-1,k}(s)}. \tag{19}$$

for the t-th transformation.

Let us first go through an example to show you when these common factors are generated and what they are composed of. Then we give a rigorous proof for their existence. Finally we talk about its applications.

4.1 Example

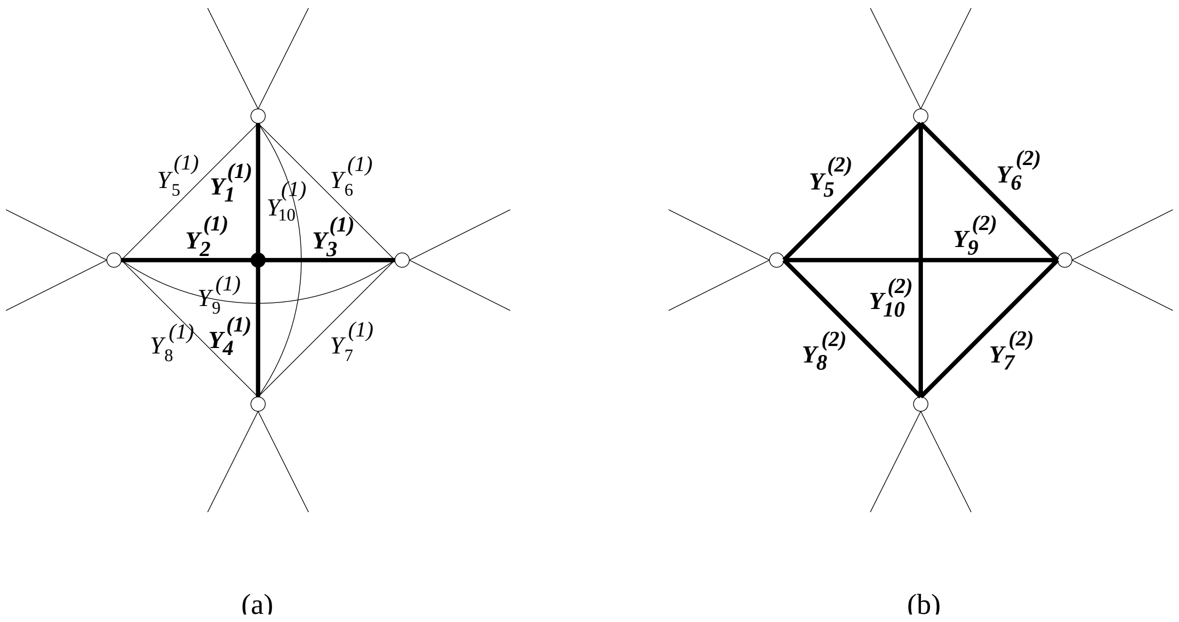


Figure 5: An example showing common factor existence—1st Y-Δ transformation: (a) original circuit schematic; (b) circuit schematic after the Y-Δ transformation.

In Fig. 5(a), we have five nodes as a portion of a circuit. We want to apply Y-Δ transformation on this portion. Literally speaking, we are going to eliminate the central solid node from the graph, and so as its four arcs. The graph after the transformation is shown in Fig. 5(b). The admittance in Fig. 5(a) is:

$$Y_1^{(1)} = \frac{a_1}{b_1},\ Y_2^{(1)} = \frac{a_2}{b_2},\ Y_3^{(1)} = \frac{a_3}{b_3},\ Y_4^{(1)} = \frac{a_4}{b_4},\ Y_5^{(1)} = \frac{a_5}{b_5},$$
$$Y_6^{(1)} = \frac{a_6}{b_6},\ Y_7^{(1)} = \frac{a_7}{b_7},\ Y_8^{(1)} = \frac{a_8}{b_8},\ Y_9^{(1)} = \frac{a_9}{b_9},\ Y_{10}^{(1)} = \frac{a_{10}}{b_{10}}.$$

Let us check out how to evaluate admittance $Y_5^{(2)}$ in Fig. 5(b). As shown in Fig. 6, $Y_5^{(2)}$ is a combination of two parallel admittance: $Y_5^{(1)}$, and $Y_5^{(2)'}$. $Y_5^{(1)}$ comes from Fig. 5(a) and $Y_5^{(2)'}$ is newly generated by $Y_1^{(1)}$, $Y_2^{(1)}$, $Y_3^{(1)}$, and $Y_4^{(1)}$:

$$\begin{aligned} Y_5^{(2)'} &= \frac{\frac{a_1}{b_1} \cdot \frac{a_2}{b_2}}{\frac{a_1}{b_1} + \frac{a_2}{b_2} + \frac{a_3}{b_3} + \frac{a_4}{b_4}} \\ &= \frac{a_1 a_2 b_3 b_4}{a_1 b_2 b_3 b_4 + b_1 a_2 b_3 b_4 + b_1 b_2 a_3 b_4 + b_1 b_2 b_3 a_4}. \end{aligned} \tag{20}$$

We define

$$\omega \equiv a_1 b_2 b_3 b_4 + b_1 a_2 b_3 b_4 + b_1 b_2 a_3 b_4 + b_1 b_2 b_3 a_4. \tag{21}$$

Then (20) can be rewritten as

$$Y_5^{(2)'} = \frac{a_1 a_2 b_3 b_4}{\omega} \equiv \frac{t_{1,2}}{\omega}. \tag{22}$$

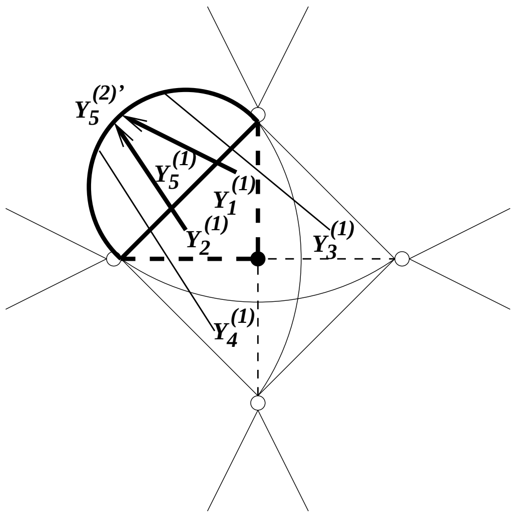


Figure 6: An example showing common factor existence—1st Y-Δ transformation : intermediate

Similarly, we have

$$\begin{aligned} Y_6^{(2)'} &= \frac{a_1a_3b_2b_4}{\omega} \equiv \frac{t_{1,3}}{\omega}, && (23)\\ Y_7^{(2)'} &= \frac{a_1a_4b_2b_3}{\omega} \equiv \frac{t_{1,4}}{\omega}, && (24)\\ Y_8^{(2)'} &= \frac{a_2a_3b_1b_4}{\omega} \equiv \frac{t_{2,3}}{\omega}, && (25)\\ Y_9^{(2)'} &= \frac{a_2a_4b_1b_3}{\omega} \equiv \frac{t_{2,4}}{\omega}, && (26)\\ Y_{10}^{(2)'} &= \frac{a_3a_4b_1b_2}{\omega} \equiv \frac{t_{3,4}}{\omega}. && (27)\end{aligned}$$

Therefore, $Y_5^{(2)}$ can be evaluated as

$$\begin{aligned} Y_5^{(2)} &= Y_5^{(1)} + Y_5^{(2)'}\\ &= \frac{a_5}{b_5} + \frac{t_{1,2}}{\omega}\\ &= \frac{a_5\omega + t_{1,2}b_5}{b_5\omega}.\end{aligned}$$

Similarly,

$$\begin{aligned} Y_6^{(2)} &= \frac{a_6\omega + t_{1,3}b_6}{b_6\omega},\ Y_7^{(2)} = \frac{a_7\omega + t_{3,4}b_7}{b_7\omega},\ Y_8^{(2)} = \frac{a_8\omega + t_{2,4}b_8}{b_8\omega},\\ Y_9^{(2)} &= \frac{a_9\omega + t_{2,3}b_9}{b_9\omega},\ Y_{10}^{(2)} = \frac{a_{10}\omega + t_{1,4}b_{10}}{b_{10}\omega}.\end{aligned}$$

Now let us apply Y-Δ transformation once again, as shown in Fig. 7(a). Admittance X'/Y' denotes the

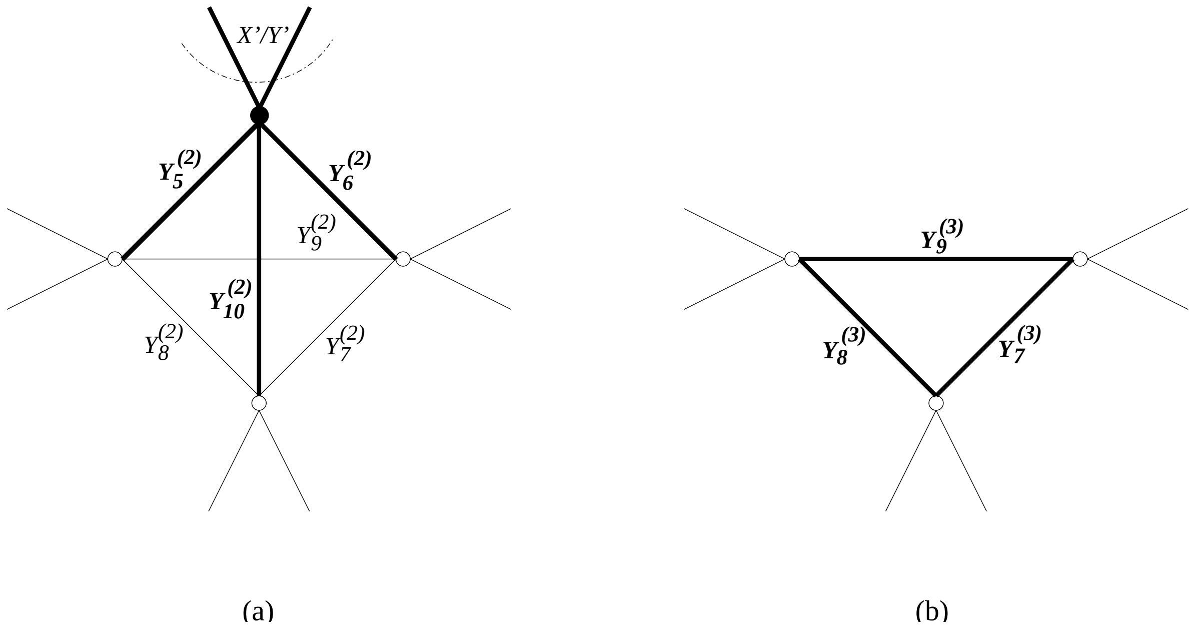


Figure 7: An example showing common factor existence—2nd Y-Δ transformation: (a) before the transformation; (b) after the transformation.

effective admittance. Let us see how to evaluate $Y_9^{(3)}$ in Fig. 7(b).

$$\begin{aligned}
Y_9^{(3)} &= Y_9^{(2)} + \frac{Y_5^{(2)}Y_6^{(2)}}{Y_5^{(2)}+Y_6^{(2)}+Y_{10}^{(2)}+\frac{X'}{Y'}} \\
&= \frac{a_9\omega+t_{2,3}b_9}{b_9\omega} + \frac{\frac{a_5\omega+t_{1,2}b_5}{b_5\omega}\,\frac{a_6\omega+t_{1,3}b_6}{b_6\omega}}{\frac{a_5\omega+t_{1,2}b_5}{b_5\omega}+\frac{a_6\omega+t_{1,3}b_6}{b_6\omega}+\frac{a_{10}\omega+t_{1,4}b_{10}}{b_{10}\omega}+\frac{X'}{Y'}} \\
&= \frac{a_9\omega+t_{2,3}b_9}{b_9\omega} + \frac{(a_5\omega+t_{1,2}b_5)(a_6\omega+t_{1,3}b_6)Y'b_{10}}{\omega\big[(a_5\omega+t_{1,2}b_5)b_6b_{10}Y'+(a_6\omega+t_{1,3}b_6)b_5b_{10}Y'+(a_{10}\omega+t_{1,4}b_{10})b_5b_6Y'+\omega b_5b_6b_{10}X'\big]} \\
&\equiv \frac{Y_{9_n}^{(3)}}{Y_{9_d}^{(3)}},
\end{aligned}$$

where

$$Y_{9_d}^{(3)} = b_9\underline{\omega}\big[(a_5\omega+t_{1,2}b_5)b_6b_{10}Y'+(a_6\omega+t_{1,3}b_6)b_5b_{10}Y'+(a_{10}\omega+t_{1,4}b_{10})b_5b_6Y'+\omega b_5b_6b_{10}X'\big],$$

and

$$\begin{aligned}
Y_{9_n}^{(3)} \;=\;& (a_9\omega+t_{2,3}b_9)\big[(a_5\omega+t_{1,2}b_5)b_6b_{10}Y'+(a_6\omega+t_{1,3}b_6)b_5b_{10}Y'+(a_{10}\omega+t_{1,4}b_{10})b_5b_6Y'+\omega b_5b_6b_{10}X'\big] \\
&+ b_9(a_5\omega+t_{1,2}b_5)(a_6\omega+t_{1,3}b_6)Y'b_{10}. \qquad (28)
\end{aligned}$$

By extending the right-hand side of (28) and reordering it to separate the terms with ω and without ω, we can rewrite $Y_{9_n}^{(3)}$ as

$$\begin{aligned}
Y_{9_n}^{(3)} &= b_9(t_{1,2}b_5)(t_{1,3}b_6)Y'b_{10}+(t_{2,3}b_9)\big[(t_{1,2}b_5)b_6b_{10}Y'+(t_{1,3}b_6)b_5b_{10}Y'+(t_{1,4}b_{10})b_5b_6Y'\big]+\omega(\cdots) \\
&= \underline{\Big\{t_{1,2}t_{1,3}+t_{2,3}\big[t_{1,2}+t_{1,3}+t_{1,4}\big]\Big\}}b_5b_6b_9b_{10}Y'+\omega(\cdots). \qquad (29)
\end{aligned}$$

Replace $t_{i,j}$ in (29) according to (22)–(27),

$$\begin{aligned}
Y_{9_n}^{(3)} &= \Big\{(a_1a_2b_3b_4)(a_1a_3b_2b_4) + (a_2a_3b_1b_4)\big[\underline{a_1}a_2b_3b_4 + \underline{a_1}a_3b_2b_4 + \underline{a_1}a_4b_2b_3\big]\Big\}b_5b_6b_9b_{10}Y' + \omega(\cdots) \\
&= \Big\{\underline{(a_1a_2a_3b_4)}(a_1b_2b_3b_4) + \underline{(a_1a_2a_3b_4)}\big[a_2b_3b_4b_1 + a_3b_2b_4b_1 + a_4b_2b_3b_1\big]\Big\}b_5b_6b_9b_{10}Y' + \omega(\cdots) \\
&= a_1a_2a_3b_4\underline{\omega}b_5b_6b_9b_{10}Y' + \underline{\omega}(\cdots). \qquad (30)
\end{aligned}$$

We have noticed that there is one ω in $Y_{9_d}^{(3)}$. The point here is that there is also one ω in $Y_{9_n}^{(3)}$, such that these two ω can be canceled. And this property is also held for the numerators of $Y_7^{(3)}$ and $Y_8^{(3)}$. The underlined parts in (31) and (32) are very similar to that in (29).

$$Y_{7_n}^{(3)} = \underline{\Big\{t_{1,3}t_{1,4} + t_{3,4}\big[t_{1,2} + t_{1,3} + t_{1,4}\big]\Big\}}b_5b_6b_7b_{10}Y' + \omega(\cdots), \qquad (31)$$

and

$$Y_{8_n}^{(3)} = \underline{\Big\{t_{1,2}t_{1,4} + t_{2,4}\big[t_{1,2} + t_{1,3} + t_{1,4}\big]\Big\}}b_5b_6b_8b_{10}Y' + \omega(\cdots). \qquad (32)$$

And actually both of them also have the same factor ω. This is not a coincidence. We will give a rigorous proof after Theorem 4.

ω is composed when the solid node in Fig. 5(a) is eliminated. And it appears in numerators of some $Y_i^{(3)}$ as well when one of the node's four neighbors is eliminated.

4.2 ω Exists in General Graphs

In this sub-section we will verify that our intuition from the example above is generally true. In other words, although the solid node in Fig. 5(a) is of degree 4, we can prove that (30) have a factor ω if the node were of degree k. This ω is the general form of (21). It is worthy noting the general meaning of ω.

Definition 1. *Given a node n_0 in a multi-port RKC linear time-invariant network, suppose n_0 has k neighbors and it is eligible for Y-Δ transformation. Denote admittance between n_0 and its neighbor n_i as $\frac{a_i}{b_i}, i \in \{1, k\}$. Particularly, we assume $b_1, \ldots, b_k$ are exclusively prime to each other. We define*

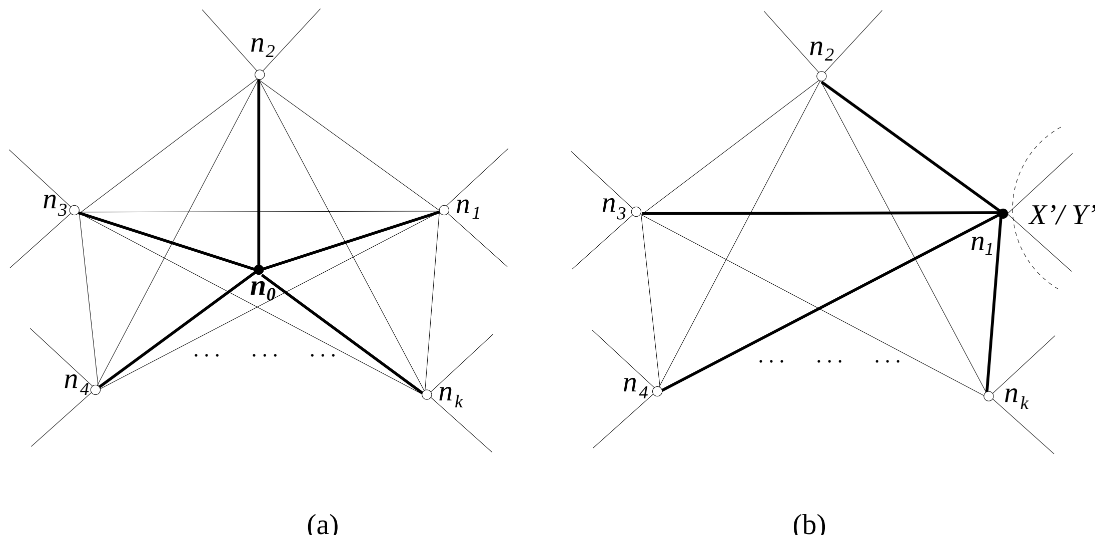
$$\omega = \sum_{i=1}^{k}\left(a_i \prod_{j=1, j\neq i}^{k} b_j\right) \qquad (33)$$

Lemma 1. *Any multi-port RKC linear time-invariant network can be represented by $G(V, E)$. $\forall n_i \in V$ with exactly four neighbors, there is an expression ω_i, which is the denominator of admittance generated when one applies Y-Δ transformation on n_i. Suppose V_i is the set of four neighbors of n_i. When n_i is eliminated and any node $n_j \in V_i$ is eliminated later, each numerator of admittance across any two nodes in $V_i - \{n_j\}$ has a multiplication factor ω_i.*

Proof: The proof is straightforward from the example in the last section. Lemma 1. can be extended to nodes with k neighbors, $k \geq 3$.

Theorem 4. *Any multi-port RKC linear time-invariant network can be represented by $G_0(V_0, E_0)$. $\forall n_i \in V$ with three or more neighbors ($k \geq 3$), there is an expression ω_i, which is the denominator of admittance generated when one applies Y-Δ transformation on n_i. Suppose L_i is the set of neighbors of n_i. When n_i is eliminated and any node $n_j \in L_i$ is eliminated later, each numerator of admittance among $L_i - \{n_j\}$ has a multiplication factor ω_i.*

Proof: For a given graph $G(V, E)$ as depicted in the theorem, we choose a node in V arbitrarily, and we denote it as n_0. Also we denote its k neighbors as $n_1, \ldots, n_k$, as shown in Fig. 8(a). We further denote admittance



n_2
n_3
n_1
n_0
n_4
n_k
$\cdots \quad \cdots \quad \cdots$
(a)
n_2
n_3
X'/Y'
n_1
n_4
n_k
$\cdots \quad \cdots \quad \cdots$
(b)

Figure 8: Illustration for proof of Theorem 4: (a) n_0 to be eliminated; (b) n_1 to be eliminated.

between any node n_i in $\{n_1, \ldots, n_k\}$ and n_0 as $Y_i^{(1)}$, and any two nodes n_i, n_j in $\{n_1, \ldots, n_k\}$ as $Y_{i,j}^{(1)}$,

$$Y_i^{(1)} = \frac{a_i}{b_i}, \quad \text{and} \quad Y_{i,j}^{(1)} = \frac{a_{i,j}}{b_{i,j}}.$$

Now let us perform Y-Δ transformation on node n_0. This process will generate a new branch between any two of n_0's neighbors, so the total number of new branches would be $\frac{k(k-1)}{2}$. The admittance of these new branches can be denoted as $Y_{i,j}^{(2)'}$. These branches will be merged with $Y_{i,j}^{(1)}$. At the moment when we are to eliminate one of n_0's neighbors, we denote the admittance between any two nodes $n_i, n_j \in \{n_1, \ldots, n_k\}$ as $Y_{i,j}^{(2)}$. According to (19), $Y_{i,j}^{(2)'}$ can be derived as follows.

$$Y_{i,j}^{(2)'} = \frac{\frac{a_i}{b_i}\frac{a_j}{b_j}}{\sum_{\nu=1}^{k} \frac{a_\nu}{b_\nu}}, \quad i, j \in \{1, \ldots, k\} \text{ and } i < j. \tag{34}$$

Let us define

$$\omega_0 \equiv \sum_{i=1}^{k} \left(a_i \prod_{j=1, j \neq i}^{k} b_j \right). \tag{35}$$

Then (34) can be written as

$$Y_{i,j}^{(2)'} = \frac{a_i a_j \prod_{\nu=1,\nu \neq i,j}^{k} b_\nu}{\omega_0}, \quad i, j \in \{1, \ldots, k\} \text{ and } i < j. \tag{36}$$

We define

$$t_{i,j} = a_i a_j \prod_{\nu=1,\nu\neq i,j}^{k} b_\nu, \quad i, j \in \{1, \ldots, k\} \text{ and } i < j, \tag{37}$$

and then we can rewrite (36) as

$$Y_{i,j}^{(2)'} = \frac{t_{i,j}}{\omega_0}, \quad i, j \in \{1, \ldots, k\} \text{ and } i < j.$$

Without losing generality, let us assume that the next node to be eliminated among n_0's neighbors is n_1. To make a rigorous proof, We can not take it for granted that

$$Y_{i,j}^{(2)} = Y_{i,j}^{(1)} + Y_{i,j}^{(2)'} \quad i,j \in \{1,\ldots,k\} \text{ and } i<j$$

because after n_0, the next node to be eliminated in the whole graph may or may *not* be n_1. if it is not, then $Y_{i,j}^{(2)}$ has to be expressed as

$$Y_{i,j}^{(2)} = Y_{i,j}^{(1)} + \frac{A}{B} + Y_{i,j}^{(2)'}, \quad i,j \in \{1,\ldots,k\} \text{ and } i<j, \tag{38}$$

where $\frac{A}{B}$ is the effective admittance that has been merged onto the admittance between n_i and n_j when some other nodes are eliminated *before* n_1 and *after* n_0. Fortunately, we do not have to care about what $Y_{i,j}$ is, as long as it is a rational function of s. So it is natural to consider $Y_{i,j}$ and $\frac{A}{B}$ as one effective admittance, and let us denote it as

$$\widetilde{Y}_{i,j}^{(1)} = Y_{i,j}^{(1)} + \frac{A}{B} \equiv \frac{\tilde{a}_{i,j}}{\tilde{b}_{i,j}}, \quad i,j \in \{1,\ldots,k\} \text{ and } i<j. \tag{39}$$

Combining (36) and (39), we can rewrite (38) as follows:

$$Y_{i,j}^{(2)} = \frac{\tilde{a}_{i,j}\omega_0 + \tilde{b}_{i,j}t_{i,j}}{\tilde{b}_{i,j}\omega_0}, \quad i,j \in \{1,\ldots,k\} \text{ and } i<j. \tag{40}$$

Now let us eliminate n_1. After we eliminate n_0, The topology of the graph in Fig. 8(a) has be changed, as shown in Fig. 8(b). Except for those belonging to the clique formed by n_0, X'/Y' denotes the effective admittance associated with n_1 outside the clique. This time let us denote the admittance between any two nodes $n_i, n_j \in \{n_2,\ldots,n_k\}$ as $Y_{i,j}^{(3)}$. According to (19), the admittance of $\frac{(k-1)(k-2)}{2}$ new branches after being merged with $Y_{i,j}^{(2)}$ can be written as

$$\begin{aligned} Y_{i,j}^{(3)} &= Y_{i,j}^{(2)} + \frac{Y_{1,i}^{(2)} Y_{1,j}^{(2)}}{\left(\sum_{\nu=2}^{k} Y_{1,\nu}^{(2)}\right) + \frac{X'}{Y'}} \\ &\equiv \frac{Y_{i,j_n}^{(3)}}{Y_{i,j_d}^{(3)}}, \qquad i,j \in \{2,\ldots,k\} \text{ and } i<j. \end{aligned} \tag{41}$$

Inserting (40) into (41), we can write $Y_{i,j_n}^{(3)}$ as

$$\begin{aligned} Y_{i,j_n}^{(3)} &= \left(\tilde{a}_{i,j}\omega_0 + \tilde{b}_{i,j}t_{i,j}\right)\left[\sum_{\nu=2}^{k}\left(\tilde{a}_{1,\nu}\omega_0 + \tilde{b}_{1,\nu}t_{1,\nu}\right)\left(\prod_{p=2,p\neq\nu}^{k} b_{1,p}\right)Y' + \omega_0\left(\prod_{\nu=2}^{k} b_{1,\nu}\right)X'\right] \\ &\quad + \tilde{b}_{i,j}\left(\tilde{a}_{1,i}\omega_0 + \tilde{b}_{1,i}t_{1,i}\right)\left(\tilde{a}_{1,j}\omega_0 + \tilde{b}_{1,j}t_{1,j}\right)\left(\prod_{\nu=2,\nu\neq i,j}^{k} b_{1,\nu}\right)Y'. \end{aligned} \tag{42}$$

Expand the right-hand side of (42) and separate terms with ω_0 and without ω_0,

$$\begin{aligned}
Y_{i,j_n}^{(3)} &= \left(\tilde{b}_{i,j}t_{i,j}\right)\left[\sum_{\nu=2}^{k}\left(\underline{\tilde{b}_{1,\nu}}t_{1,\nu}\right)\underline{\left(\prod_{p=2,p\neq\nu}^{k} b_{1,p}\right)}Y'\right] + \tilde{b}_{i,j}\left(\underline{\tilde{b}_{1,i}}t_{1,i}\right)\left(\underline{\tilde{b}_{1,j}}t_{1,j}\right)\underline{\left(\prod_{p=2,p\neq i,j}^{k} b_{1,p}\right)}Y' + \omega_0(\cdots) \\
&= \left(\tilde{b}_{i,j}t_{i,j}\right)\left[\sum_{\nu=2}^{k} t_{1,\nu}\right]\left(\prod_{p=2}^{k} b_{1,p}\right)Y' + \tilde{b}_{i,j}t_{1,i}t_{1,j}\left(\prod_{p=2}^{k} b_{1,p}\right)Y' + \omega_0(\cdots) \\
&= \underline{\left(t_{i,j}\sum_{\nu=2}^{k} t_{1,\nu} + t_{1,i}t_{1,j}\right)}\tilde{b}_{i,j}\prod_{p=2}^{k} b_{1,p}Y' + \omega_0(\cdots) \qquad (43)\\
&\equiv T_{i,j}\tilde{b}_{i,j}\prod_{p=2}^{k} b_{1,p}Y' + \omega_0(\cdots).
\end{aligned}$$

As we mentioned, (29), 31 and 32 are just special cases of (43). Replace $t_{i,j}$ in (43) according to (37), the underlined part can be rewritten as

$$\begin{aligned}
T_{i,j} &= \left(a_i a_j \prod_{\nu=1,\nu\neq i,j}^{k} b_\nu\right)\left[\sum_{\nu=2}^{k}\left(a_1 a_\nu \prod_{p=1,p\neq 1,\nu}^{k} b_p\right)\right] + \left(a_1 a_i \prod_{\nu=1,\nu\neq 1,i}^{k} b_\nu\right)\left(a_1 a_j \prod_{\nu=1,\nu\neq 1,j}^{k} b_\nu\right) \\
&= a_1\left(a_i a_j \prod_{\nu=1,\nu\neq i,j}^{k} b_\nu\right)\left[\sum_{\nu=2}^{k}\left(a_\nu \prod_{p=1,p\neq 1,\nu}^{k} b_p\right)\right] + a_j\left(a_1 a_i \prod_{\nu=1,\nu\neq 1,i}^{k} b_\nu\right)\left(a_1 \prod_{\nu=1,\nu\neq 1,j}^{k} b_\nu\right) \\
&= a_1\left(a_i a_j \prod_{\nu=2,\nu\neq i,j}^{k} b_\nu\right)\left[b_1\sum_{\nu=2}^{k}\left(a_\nu \prod_{p=1,p\neq 1,\nu}^{k} b_p\right)\right] + a_j\left(a_1 a_i \prod_{\nu=1,\nu\neq 1,i,j}^{k} b_\nu\right)\left(b_j a_1 \prod_{\nu=1,\nu\neq 1,j}^{k} b_\nu\right) \\
&= \left(a_1 a_i a_j \prod_{\nu=2,\nu\neq i,j}^{k} b_\nu\right)\left[b_1\sum_{\nu=2}^{k}\left(a_\nu \prod_{p=1,p\neq 1,\nu}^{k} b_p\right) + b_j a_1 \prod_{\nu=1,\nu\neq 1,j}^{k} b_\nu\right] \\
&= \left(a_1 a_i a_j \prod_{\nu=2,\nu\neq i,j}^{k} b_\nu\right)\omega_0. \qquad (44)
\end{aligned}$$

So overall speaking, $Y_{i,j_n}^{(3)}$ has a factor ω_0.

Please note that when we were rewriting (34) into (36), we actually simplified the scenario because we took it for granted that each b_ν in (34) is relatively prime to each other. Unfortunately, to achieve the optimal result for both efficiency and accuracy, we can not assume so. For example if $b_1 = t_0t_1$ and $b_2 = t_0t_2$, then we have to rewrite the denominator in (34) as

$$\frac{a_1}{t_0t_1} + \frac{a_2}{t_0t_2} + \sum_{\nu=3}^{k}\frac{a_\nu}{b_\nu} = \frac{a_1t_2 + a_2t_1}{t_0t_1t_2} + \sum_{\nu=3}^{k}\frac{a_\nu}{b_\nu}. \qquad (45)$$

In our simplified scenario, we have rewritten it as

$$\frac{a_1}{t_0t_1} + \frac{a_2}{t_0t_2} + \sum_{\nu=3}^{k}\frac{a_\nu}{b_\nu} = \frac{a_1t_0t_2 + a_2t_0t_1}{t_0^2t_1t_2} + \sum_{\nu=3}^{k}\frac{a_\nu}{b_\nu}.$$

As you can imagine, in order to consider this common t_0, we will have to rewrite ω_0 defined in (35). Let us called it ω_0'. Without the additional t_0, ω_0' is simpler than ω_0. Although we have changed ω_0 to ω_0', $Y_{i,j_n}^{(3)}$ will have a factor ω_0'. Because each multiplication term in (35) has either b_1 or b_2 or both of them as factors. So ω_0' is actually a factor of ω_0. In our example above, $\omega_0 = t_0\omega_0'$.

In the theorem we only considered nodes with more than two neighbors, as for a node with exactly two neighbors, there will be only one new branch and its admittance does not have redundant common factor in its numerator or denominator. Please note that we do not consider circuit cases with dangling or isolated nodes (with one neighbor or no neighbor at all). ■

In this theorem, we evaluated $Y^{(2)}$ from $Y^{(1)}$ and $Y^{(3)}$ from $Y^{(2)}$. In the evaluation process, we did not change $Y^{(2)}$. But if we go on the process and use $Y^{(3)}$ to evaluate $Y^{(4)}$, then before doing this, we may need to simplify some of $Y^{(3)}$ because their numerator and denominator have one or more common factors. The next theorem

$$Y^{(1)} \Rightarrow Y^{(2)} \Rightarrow Y^{(3)} \to \widetilde{Y}^{(3)} \Rightarrow Y^{(4)}$$

Figure 9: General Scenario

claims that numerators of $Y^{(4)}$ would have the ω generated by $Y^{(2)}$ even with the simplification from $Y^{(3)}$ to $\widetilde{Y}^{(3)}$ in between(Fig. 9).

THEOREM 5. *Given a multi-port RKC linear time-invariant network represented by $G_0(V_0, E_0)$, we can use a series of Y-Δ transformations to eliminate all its internal nodes. Suppose it has n internal nodes, then after we apply Y-Δ transformation on one node in $G_k(V_k, E_k)$(Fig. 10(a)), the graph will be updated and denoted as $G_{k+1}(V_{k+1}, E_{k+1})$, where $0 \le k < n-1$.*

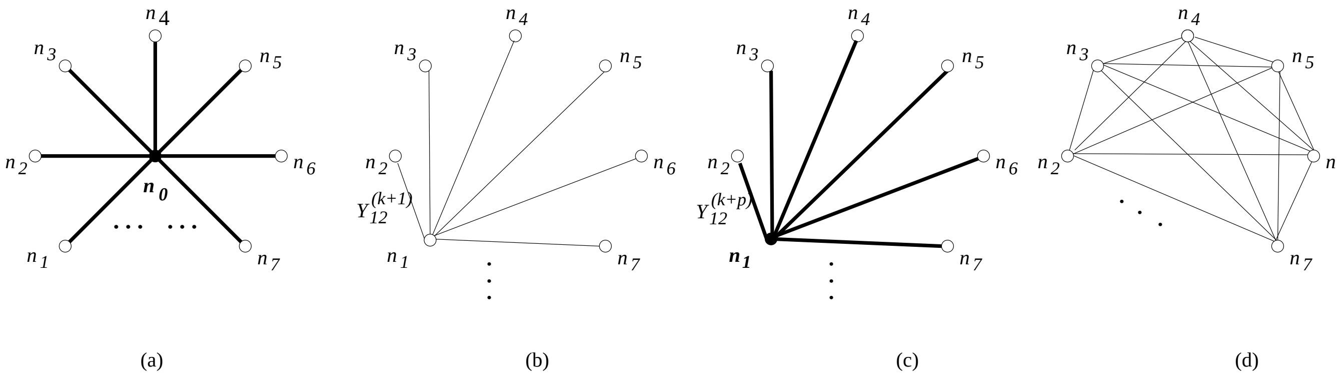


Figure 10: Illustration for Theorem 5: (a) k-th transformation; (b) $(k+1)$-th transformation; (c) $(k+p)$-th transformation; (d) $(k+p+1)$-th transformation.

Suppose n_0 is eliminated at the k-th step. Let L_0 represent the neighbors of n_0 in $G_k(V_k, E_k)$. Then we further suppose that $n_1 \in L_0$ is the first one in L_0 being eliminated after n_0. Let us say n_1 is eliminated in the $k+p$-th step. Fig. 10 shows the scenario. $Y^{(k)}$ represents the admittance between n_0 and its neighbors. $Y^{(k+p)}$ the admittance between any two nodes in L_0, and $Y^{(k+p+1)}$ the admittance between any two nodes in $L_0 - \{n_1\}$.

numerators of $Y^{(k+p+1)}$ would have the ω_0 generated at the k-th step, no matter how many other common factors are found and canceled out during the $(k+1)$-th step through the $(k+p)$-th step.

Proof: Firstly, we know that *omega*'s associated to each internal node during the reduction process are prime to each other, as no any two nodes have exactly the same branches at any reduction steps. Unless ultimately, there remain only two nodes, when the reduction terminates.

Secondly, it is true that if we choose to postpone all the common-factor-cancellation operations during the $(k+1)$-th step through the $(k+p)$-th step, these common factors are still common factors in *all* of the admittance in $Y^{(k+p+1)}$, If we suppose ω is a common factor of $Y_{12}^{(k+1)}$ in Fig. 10(b), for instance, then it is still a common factor of $Y_{12}^{(k+p)}$ in Fig. 10(c), as only admittance-addition operations might be performed on branch between n_1

and n_2 during the $(k+p+2)$-th step through the $(k+p)$-th step. According to (15), admittance between any two nodes in $L_0 - \{n_1\}$ in the $(k+p+1)$-th step(Fig. 10(d)) has ω as its common factor. And because ω is prime to ω_0, ω_0 would still come up as a common factor to each admittance in $Y^{(k+p+1)}$ if we had canceled out ω in earlier steps. ■

Similar to mathematical induction, Theorem 4 assures the foundation of our reduction algorithm, and theorem 5 makes our reduction process work recursively. The two theorems together support our algorithm in the next section.

4.3 Common Factors in Denominators Only

There is another kind of common factors: for admittance of two branches that were connected to the same node n_x eliminated earlier, the denominators of the two admittance share the ω associated to n_x. Fig. 11 shows an example scenario of our explanation. After n_0 in (a) is eliminated, $Y_{12}^{(k+1)}$ and $Y_{13}^{(k+1)}$ in (b) share a common

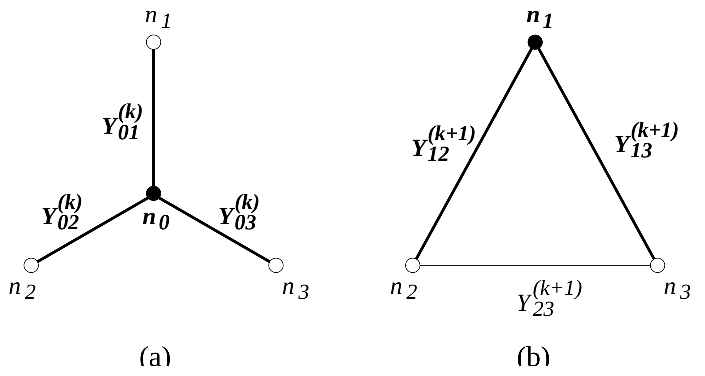


Figure 11: An example shown common factors in denominators: (a) n_0 to be eliminated; (b) n_1 to be eliminated.

factor ω from n_0. And when n_1 is to be eliminated using (15, the common factor in the denominator of the right-hand side has to be identified. Please note that this ω is not necessarily equal to the whole denominator of $Y_{12}^{(k+1)}$ or $Y_{13}^{(k+1)}$, but generally serves as a common factor of the two.

5 Overall Algorithm

Collecting all the rules that we have had so far for Y-Δ transformation, we give an algorithm in pseudo-code. Given a multi-port RKC linear time-invariant network, one can represent it as a graph $G_0(V_0, E_0)$, with $\tilde{V}$ including all external nodes.

Algorithm 1.

```
1. order nodes in $V_0 - \tilde{V}$ and generate a node elimination sequence $S$;
2. for each node $n_i \in V_0 - \tilde{V}$, create a set $L_i$ and initiates its value as its neighbors in $G_0$;
3. for each node $n_i \in V_0 - \{n_p, n_q\}$, create a set $n_i.prev$ and $n_i.prev = \phi$;
4. for $n_i = S[i]$ do
    4.1. find the node $n_k$ in $L_i$ that appears first in $S$;
    4.2. $n_k.prev = n_k.prev \cup \{n_i\}$;
```

```
4.3. for $n_j \in n_i.prev$ do:
    4.3.1. $t_j = |L_i \cap L_j| - 1$;
4.4 perform Y-$\Delta$ transformation on node $n_i$:
    4.4.1. denominator: $Y_{i_d}^{(i)} = \sum_{n_j \in L_i} Y_{i,j}^{(i-1)}$;
    4.4.2. remove redundancy: divide the numerator and denominator of $Y_{i_d}^{(i)}$ with $\prod_{n_j \in n_i.prev} \omega_j^{t_j}$, then $Y_{i_d}^{(i)} \equiv \frac{\omega_i}{A}$;
    4.4.3. for $n_j, n_k \in L_i,\ j \neq k$ do:
        4.4.3.1. $\Omega = 1.0$;
        4.4.3.2. for $n_\nu \in n_i.prev$ do:
            4.4.3.2.1. if $n_j \in L_\nu$ and $n_k \in L_\nu$, then $\Omega = \Omega \cdot \omega_\nu$;
        4.4.3.3. numerator: $Y_{i_n}^{(i)} = Y_{j,i}^{(i-1)} Y_{i,k}^{(i-1)} \equiv \frac{B}{C}$;
        4.4.3.4. new admittance: $Y_{j,k}^{(i)'} = \frac{Y_{i_n}^{(i)}}{Y_{i_d}^{(i)}} = \frac{\frac{B}{C}A}{\omega_i}$;
        4.4.3.5. remove redundancy: $Y_{j,k}^{(i)'} = \frac{B\frac{A\Omega}{C}}{\Omega\omega_i} \equiv \frac{D}{E}$, where $\frac{A\Omega}{C}$ is a division operation;
        4.4.3.6. merging, if necessary: $Y_{j,k}^{(i)} = Y_{j,k}^{(i)'} + Y_{j,k}^{(i-1)} = \frac{D}{E} + \frac{G}{H} = \frac{DH+GE}{EH}$, where $Y_{j,k}^{(i-1)} = \frac{G}{H}$;
        4.4.3.7. remove redundancy: if $\Omega \neq 1.0$, then $Y_{j,k}^{(i)} = \frac{(DH+GE)/\Omega^2}{EH/\Omega^2}$;
    4.4.4. $G_{i-1}$ has been transformed to $G_i$;
4.5. update $L_j$ of $n_j \in L_i$.
```

Topological analysis method is another approach to obtaining the driving-point admittance functions by evaluating determinants and cofactors of admittance matrices. One advantage of topological formulas over the conventional methods for the evaluation of determinants and cofactors is that the former avoids the usual cancellations inherent in the expansion of determinants and cofactors in the latter. The determinant of the node admittance matrix of a passive network without mutual inductances is equal to the sum of all the tree admittance products of it. The enumeration of all the trees of a graph is very time-consuming[27]. The approach is generally an exponential algorithm in terms of time. In the worst case when G_0 is a complete graph, there are $O(N^{N-2})$ trees in the graph and evaluating all the admittance products takes an exponential amount of time, where N is the number of nodes in G_0.

For a given graph G_0, we assume that N is the number of nodes in G_0, d is the maximum degree of nodes in G_0, and $\tilde{d}$ is the overall maximum degree of nodes G_0, G_1, and so on. Of course $\tilde{d} > d$. r is supposed to be the number of orders reserved for each admittance, and t the maximum size of $n_i.prev$ defined first in Step 3.

In our algorithm, Step 1 has $O(N \cdot \tilde{d})$ operations, even without mass elimination. Step 2 takes $O(N \cdot d)$ time, and Step 3 $O(N)$. Step 4 itself iterates about N times, assuming the number of external nodes is far less than N. Inside each iteration, 4.1 takes $O(1)$ time because we always maintain the list L of every node along with new branch generation. 4.2 also takes $O(1)$ time. As each node can only be added into another node's *prev* list once, 4.3 takes $O(\tilde{d})$ time in average in every iteration of Step 4. Inside 4.4, 4.4.1 takes $O(\tilde{d} \cdot r^2)$, because each polynomial product operation takes $O(r^2)$ and it dominates any polynomial addition operation. Because the maximum value of t_j from 4.3.1 would be $\tilde{d}$. 4.4.2 takes $O(\tilde{d} \cdot r^2)$ time in average in every iteration of Step

4, for the similar reason for 4.3. 4.4.3 iterates $O(\tilde{d}^2)$ times in each iteration of Step 4. Inside it, 4.4.3.1 takes constant time, and 4.4.3.4 takes no time, as it is only a statement. 4.4.3.2 takes $O(r^2 \cdot t)$ time. Note that the condition evaluation takes constant time, because neighbors of $n_i.prev$ was scanned in 4.3 and stored for late use. 4.4.3.5–4.4.3.7 each takes $O(r^2)$ time. Finally, 4.5 takes $O(\tilde{d}^2)$. So overall speaking, 4.4 takes $O(\tilde{d}^2 \cdot r^2 \cdot t)$ and hence Step 4 takes $O(N \cdot \tilde{d}^2 \cdot r^2 \cdot t)$.

The worst case happens when G_0 is a complete graph. $\tilde{d} = N - 1$ and $t = 1$. So the worst case complexity is $O(N^3 \cdot r^2)$.

Different from LU decomposition in SPICE, our algorithm enables dynamical memory de-allocation, as branches of nodes eliminated are no longer useful and the memory thus can be freed. As a result, the memory requirement grows up in the middle of the reduction process and goes down till the end of it. For graphs derived from VLSI circuits, the proposed algorithm is proportional to the overall maximum number of branches in G_0, G_1, etc. The worst case happens when G_0 is a complete graph and the complexity is $O(N^2)$.

6 Admittance in Its Simplest Form

If we apply Algorithm 1 on complete graphs, the final solution is optimal, meaning it does not has any common factors and thus it has the same order as the original network. Let us first look at a 6-node complete graph. Its transformation process is shown in Fig. 12. Solid nodes in the figure are to be eliminated at the snapshot. Y_i denotes the form of admittance in each step, where denominators are given and numerators are simply ignored (*). Please keep in mind that these numerators have the same order as their correspondent denominators.

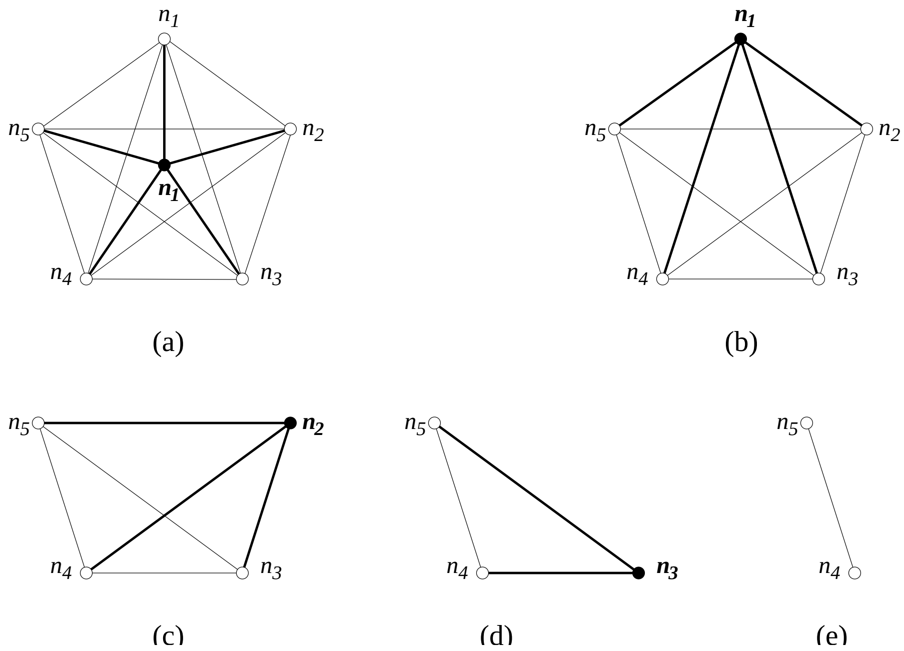


Figure 12: (a) $Y_i = \frac{a_i}{b_i}$, $|\omega_1| = 5$; (b) $Y_i = \frac{*}{b_i\omega_1}$, $|\omega_2| = 9$; (c) $Y_i = \frac{*}{b_i\omega_2}$, $|\omega_3| = 12$; (d) $Y_i = \frac{*}{b_i\omega_3}$, $|\omega_4| = 14$; (e) $Y_i = \frac{*}{b_i\omega_4}$

In Fig. 12(a), five heavily weighted branches are going to be eliminated along with the central node. New branches are in the form of

$$\frac{\frac{a_i}{b_i}\frac{a_j}{b_j}}{\frac{a_1}{b_1} + \frac{a_2}{b_2} + \frac{a_3}{b_3} + \frac{a_4}{b_4} + \frac{a_5}{b_5}} = \frac{*}{\frac{a_1b_2b_3b_4b_5 + \ldots + b_1b_2b_3b_4a_5}{b_1b_2b_3b_4b_5}} \equiv \frac{*}{\frac{\omega_1}{b_1b_2b_3b_4b_5}} = \frac{*}{\omega_1} \tag{46}$$

When these new branches are to be combined with those existing in parallel

$$\frac{*}{\omega_1} + \frac{a_i}{b_i} = \frac{*}{b_i\omega_1}, \tag{47}$$

which constitute Y_is in Fig. 12(b).

Let us go one step further. When we are eliminating the second node(Fig. 12(b)), new branches are in the form of

$$\frac{\frac{*}{b_i\omega_1}\frac{*}{b_j\omega_1}}{\frac{*}{b_6\omega_1}+\frac{*}{b_7\omega_1}+\frac{*}{b_8\omega_1}+\frac{*}{b_9\omega_1}} \equiv \frac{\frac{*}{b_ib_j\omega_1^2}}{\frac{\omega_2}{b_6b_7b_8b_9\omega_1}} = \frac{*}{\omega_1\omega_2}. \tag{48}$$

After the similar step as 47, the final admittance Y_i in Fig. 12(c) can be evaluated as

$$\frac{*}{\omega_1\omega_2} + \frac{*}{b_i\omega_1} = \frac{*}{b_i\omega_1\omega_2}. \tag{49}$$

Due to Theorem 4, ω_1 in (49) can be canceled. So in Fig. 12(c),

$$Y_i = \frac{*}{b_i\omega_2} \tag{50}$$

So on and so forth, when the network is reduced into one port(Fig. 12(e)), the order of the admittance is $|b_i\omega_4| = 15$, which is equal to the number of branches in Fig. 12(a). Note that $|\omega_2| - |\omega_1| = 4, |\omega_3| - |\omega_2| = 3, |\omega_4| - |\omega_3| = 2$.

THEOREM 6. *Given a linear RKC time-invariant network which can be represented by a graph $G(V,E)$, after Algorithm 1 reduces G into one branch whose admittance has the same order as the original network.*

Proof: Let us look at an example along our proof. Fig. 13 illustrate a Y-Δ reduction series. Solid nodes shown in each graph are the ones to be eliminated at that step. As we know, each time when we eliminate a node, there is an ω associated with it. And the denominator of the admittance associated with every branch is a multiplication of b and some of existing ω's from nodes we have eliminated so far. If the branch exists in the original graph $G_0(V_0,E_0)$, then b is the denominator of its admittance; if the branch does not exist in $G_0(V_0,E_0)$, then $b=1$. We assume each branch in E_0 has different admittance and we assign a distinct integer to each of them (Fig. 13(a)). Graphs (a)–(b) in the elimination series are denoted as $G_0(V_0,E_0), G_1(V_1,E_1),,\ldots,G_n(V_n,E_n)$, respectively, where $n = |V_0| - 2$. Along with each branch in each graph is a set S, where each member W is a set of branches. Here $W \subseteq \{1,2,\ldots,|E_0|\}$.

DEFINITION 2. *We define that a branch e is <u>associated with</u> a branch e' if there exists a W such that $W \in S$ of e' and $e \in W$, where $e \in E_0$ and $e' \in \bigcup E_i$ $(0 \le i \le |E_0|)$.*

For instance, branch a between n_1 and n_2 in Fig. 13(a) is associated with the branch between n_2 and n_8 in (b), because $a \in \{a,b\}$ in $\{\{a,b\}\}$. a is also associated with the branch between n_8 and n_9 in (c), because $a \in \{a,b,c,d,i\}$ in $\{\{k\},\{a,b,c,d,i\},\{d,e,f,g,j\}\}$.

We now give four claims following Algorithm 1.

1. When a node is to be eliminated,
   (a) $W = \bigcup_{S_i \text{ incident to the node}} \bigcup_{W_j \in S_i} W_j$;
   (b) W needs to be inserted into S of branches between any two of the node's neighbors. Particularly, if $\exists W_i \in S$ and $W_i \subset W$ before the insertion, then W_i has to be removed from the set S after that.
2. For S of *any* branch, $\forall W_i, W_j \in S \Rightarrow W_i \cap W_j = \phi$.
3. For S of *any* two *different* branches, S_1 and S_2, $\forall W_i \in S_1$ and $\forall W_j \in S_2, \Rightarrow W_i = W_j$ or $W_i \cap W_j = \phi$.
4. For S of any branch, $\sum_{W_i \in S} |W_i|$ is equal to the number of branches in E_0 that have associated with the branch.

If the four claims are true, then Theorem 6 is true automatically because finally all branches in E_0 will be associated with the ultimate branch and the order of its admittance is the same as that of the original network represented by $G_0(V_0, E_0)$.

Referring to 4.4 in Algorithm 1 on how a new admittance is evaluated, one can find the direct correspondence of 4.4.1–4.4.2 to Claim 1(a), and 4.4.3.6–4.4.3.7 to Claim 1(b). So Claim 1 is true. Claim 2 and Claim 3 are directly derived from Claim 1(b), as long as each branch in E_0 is initially assigned a unique index.

Claim 4 can be proved using mathematical induction. It is true when we apply Y-Δ transformation on some node of the original graph $G_0(V_0, E_0)$. For instance, in Fig. 13(a), four nodes at the four corners are eliminated and four new diagonal branches are generated in Fig. 13(b). The order of denominators of these new admittance (also the order of the admittance) are equal to 2, the number of branches in E_0(Fig. 13(a)) associated with each of them. With assuming Claim 3 is true for the first k steps, then Claim 1 and Claim 2 together guaranteed that it is also true for the the $k+1$-th step.

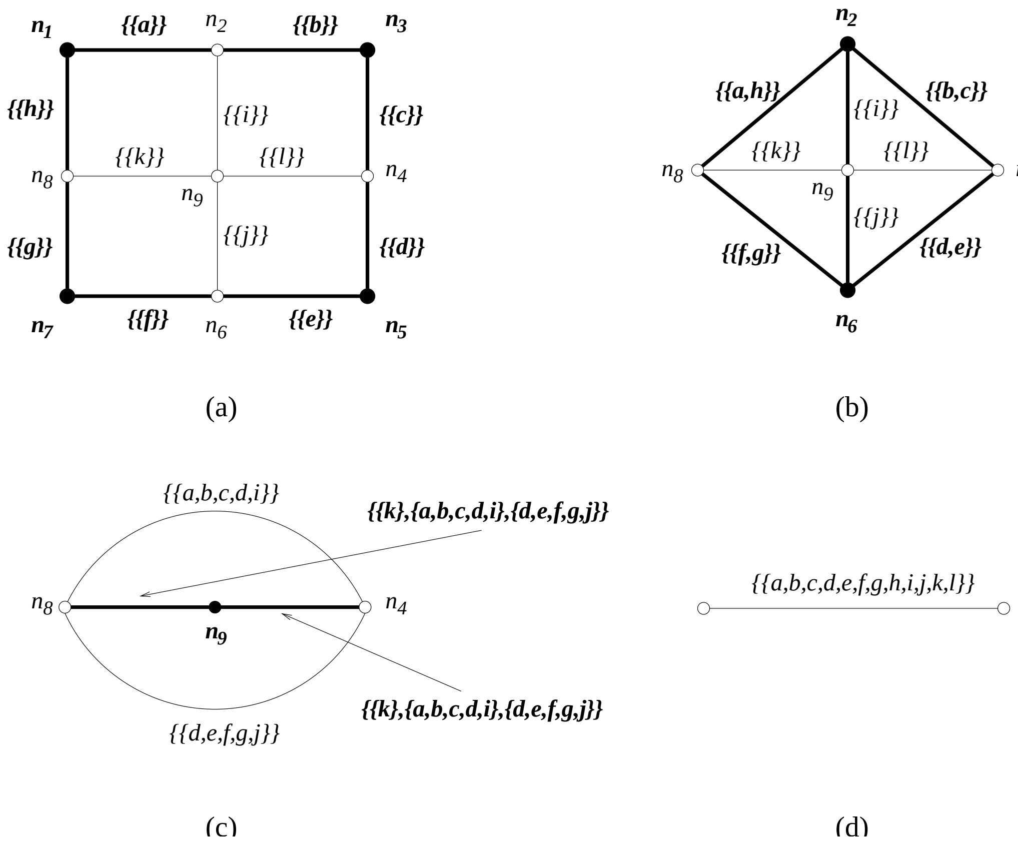


Figure 13: Illustration for proof of Theorem 6: (a) the four corner nodes are to be eliminated; (b) the two upper and lower nodes are to be eliminated; (c) the central node is to be eliminated; (d)transformation finished.

To understand this, we have to refer to Y-Δ elimination process in Algorithm 1. When generating a new admittance, we face only two cases: (1) S of each branch incident to the eliminated node is disjoint with each other, i.e. $|L_i \cap L_j| < 2$ in 4.3.1; (2) two or more S are not disjoint so that the corresponding denominators have some common factor(s), i.e. $|L_i \cap L_j| \geq 2$. We have setup some mechanism for this case in 4.4.2 to remove the redundancy. When merging is necessary, we also face two scenarios: (1) the old branch is a member of the clique generated when n_ν was eliminated. We have setup some mechanism for this case in 4.4.3.7 to remove the redundancy. Also this removal is guaranteed by Theorem 4. (2)the old branch is not a member of the clique generated when n_ν was eliminated. . So overall, the order of effected branches is still consistent with the number of branches in E_0 associated with them because of the redundancy eliminations.

This completes our proof. ■

7 Conclusion

We proposed a generalized Y-Δ transformation for interconnect model reduction. This report covered the theoretical foundation of our work. The proposed algorithm can handle linear(ized) independent sources, resistors, capacitors, self and mutual K elements. The algorithm integrated common-factor-cancellation operations that were not seen in the literature. Admittance in reduced circuits has the guaranteed simplest form. Further applications to the work can be found reported in [1].

References

[1] Zhanhai Qin, Chung-Kuan Cheng, "Linear network reduction using generalized Y-Δ transformation — applications," Technical Report, University of California, San Diego. May, 2002.

[2] A. Dharchoudhury et. al., "Design and analysis of power distribution networks in PowerPC^{TM} microprocessors," in *DAC*, pp.738–43, 1998.

[3] G. Steele et. al., "Full-chip verification methods for DSM power distribution systems," in *DAC*, pp.744–9, 1998.

[4] L.W. Nagel, "SPICE2: a computer program to simulate semiconductor circuits," Berkeley, Calif. : Electronics Research Laboratory, College of Engineering, University of California, 1975.

[5] Z. Qin, Z. Zhu, and C.K. Cheng, "Efficient transient analysis for large linear networks," in *SASIMI*, pp.293–00, 2001.

[6] T.H. Chen, C.P. Chen, " Efficient large-scale power grid analysis based on preconditioned Krylov-subspace iterative methods," in *DAC*, pp.559–62, 2000.

[7] J.W. Demmel, J.R. Gulbert, X.S. Li, "SuperLU user's guide," *National Energy Research Scientific Computing Center,* http://www.nersc.gov/ xiaoye/SuperLU, 1999

[8] S. R. Nassif, J. N. Kozhaya, "Fast power grid simulation," in *DAC*, pp.156–61, 2000.

[9] J. N. Kozhaya, S. R. Nissif, and F. N. Najm, "Multigrid-like technique for power grid analysis," in *ICCAD*, pp.480–7, 2001.

[10] L.T. Pillage and R.A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. on CAD*, vol. CAD–9, pp.352–66, Apr. 1990.

[11] S. Lin, and E. S. Kuh, "transient simulation of lossy interconnets based on the recursive convolution formulation," *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, vol.39, pp.879–92, Nov. 1992.

[12] H. Liao, W. Dai, R. Wang, and F. Y. Chang, "S-parameter based macro model of distributed-lumped networks using exponentially decayed polynomial function," in *DAC*, pp.726–31, 1993.

[13] C. L. Ratzlaff, L. T. Pillage, "RICE: rapid interconnect circuit evaluation using AWE," *IEEE Trans. on CAD*, vol 13, pp.763–6, Jun. 1994.

[14] M. Sriram, S. M. Kahng, "fast approximation of the transient response of lossy transmission line trees," in *DAC*, pp.691–6, 1993.

[15] M. Sriram, S. M. Kahng, "Performance driven MCM routing using a second order RLC tree delay model," *Proceedings. fifth Annual IEEE International Conference on Wafer Scale Integration*, pp.262–7, 1993.

[16] H. Liao, W. W.M. Dai, "Partitioning and reduction of RC interconnect networks based on scattering parameter macromodels," in *DAC*, pp.704–9, 1995.

[17] P. Feldmann, R. W. Freund, "Reduced-order modeling of large linear subcircuits via a block Lanczos algorithm," in *DAC*, pp.376-80, 1995.

[18] D. L. Boley, "Krylov space methods on state-space control models," in *Circuits Systems and Signal Processing*, vol. 13, no.6, pp.733-58, 1994.

[19] A. Odabasioglu, M. Celik, L. T. Pillage, "PRIMA: passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. on CAD*, vol.17, pp.645–54, Aug. 1998.

[20] K. J. Kerns, I. L. Wemple, A. T. Yang, "Stable and efficient reduction of substrate model networks using congruence transforms," in *ICCAD*, pp. 207-14, 1995.

[21] K. J. Kerns, "Accurate and stable reduction of RLC networks using split congruence transformations," Ph.D. Dissertation, Univ. Washington, Sept. 1996.

[22] A. Devgan, H. Ji, W. Dai, "How to efficiently capture on-chip inductance effects: introducing a new circuit element K," in *ICCAD*, pp.150-5, 2000.

[23] H. Ji, A. Devgan, W. Dai, "KSim: a stable and efficient RKC simulator for capturing on-chip inductance effect," in *ASP-DAC*, pp.379-84, 2001.

[24] A. George, J. W.H. Liu, "computer solution of large sparse positive definite systems," *Prentice-Hall*, Englewood Cliffs, NJ, 1981.

[25] J. W.H. Liu, "modification of the minimum-degree algorithm by multiple elimination," *ACM Trans. Math Software*, pp.337-58, vol.11, issue 5, Jun. 1985.

[26] M. Yannakakis, "computing the minimum fill-in is NP-complete," *SIAM J. Alg. Disc. Math*, 1, 77-9, 1981.

[27] Shu-Park Chan, "Introductory topological analysis of electrical networks," *Holt, Rinehart and Winston, Inc.*, 1974.