申请上海交通大学硕士学位论文

# 树状 RC 电路的符号化矩量计算与时序估计应用


**专　　业：** 电路与系统

**硕 士 生：** 徐迪

**导　　师：** 施国勇（教授）

**上海交通大学微电子学院**


**2008 年 11 月**


此研究由上海市浦江人才基金资助(No. 07pj14053)

**Master Dissertation Submitted to Shanghai Jiao Tong University**

# Symbolic Moment Calculation of RC Tree Circuit and its Application of Delay Metrics

**Author:** Xu, Di

**Advisor:** Prof. Shi

**Specialty:** Circuit and System

School of MicroElectronics

Shanghai Jiao Tong University

November 20, 2008

此研究由上海市浦江人才基金资助(No. 07pj14053)

# 上海交通大学
# 学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保密□，在_____年解密后适用本授权书。

本学位论文属于

不保密□。

（请在以上方框内打“√”）

学位论文作者签名：　　指导教师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

此研究由上海市浦江人才基金资助(No. 07pj14053)

# 上海交通大学
# 学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期：　　年　　月　　日

此研究由上海市浦江人才基金资助(No. 07pj14053)

# 树状 RC 电路的符号化矩量计算与时序估计应用

## 摘 要

随着集成电路领域芯片加工工艺技术向深亚微米领域发展,特征尺寸变得越来越小,模型降阶技术在电路模拟中的作用也越来越重要。随着 L.T.Pillage 提出的 AWE 算法对模型降阶技术的研究，模型降阶技术逐渐成为近年来在电路问题研究中的重要课题之一。而其中，矩对于模型降阶技术的求解起到了关键作用，迅速有效的得到电路矩就能有效的完成的模型降阶问题。另一方面，由于一阶矩和 Elmore 延时的特殊关系，使得矩在估计电路时序估计方面也起到了极为重要的作用，而为了更精确的得到电路的时序，不少学者已用 2 阶甚至 3 阶矩来近似延时。

由于电路矩的重要性,已有很多各方面关于矩的研究。本文针对典型的 RC 树状电路，通过与二叉判定图算法（BDD）和图的拓扑结构的研究，提出了一种精确快速得到电路各阶矩量的方法，从而提高了模型降阶技术的效率。在本文中，利用这种符号化计算矩的方法，验证了在估计时序方面的有效性。还通过具体分析矩的表达式来测试矩对于电路中电阻、电容以及各点前一阶矩的敏感性。

**关键词：**模型降阶；矩；BDD 算法；时序估计

# Symbolic Moment Calculation of RC Tree Circuit and its Application of Delay Metrics

## ABSTRACT

With the development of the chip processing technology of integrated circuits, the feature size become smaller, and Model Order Reduction (MOR) technology gradually plays a more important role in the circuit simulation. As L.T.Pillage made a method named AWE (Asymptotic waveform Evaluation), MOR gradually become one of the most important research topics in Circuit Design area in the recent years. Since the circuit moment plays a key role in AWE, one can finish the MOR problem efficiently if he could get moment efficiently. In the other hand, moment also plays an important role in estimating the delay because of the relationship between the 1st order moment and Elmore Delay. In order to get more accurate delay, someone had used 2nd order moment or 3rd order moment to approximating the delay.

As mentioned above, circuit moment is so important that many researches are focused on it. And in this article, proposed a new method to deal with the typical RC tree circuit by combining BDD (Binary Decision Diagram) algorithm and topology. By using this method, one could get each order moments fast and accurately. And in this article, the efficiency of the method in estimating delay is also verified. And in the end, the moment sensitivity is analyzed, too.

**Keywords:** Model Order Reduction, Moment, BDD Algorithm, Delay

# 目 录

# 第一章 概 述

## 1.1 引言

随着集成电路领域芯片加工工艺技术向深亚微米领域发展，特征尺寸变得越来越小，从0.18微米到0.13微米，到现在的90纳米，65纳米甚至将来的45纳米，互连线的耦合电容、电感对电路的影响越来越大，如何正确分析这些寄生参数对于整个电路所起的影响成为了众多电路设计人员不可回避的问题。为了能有效地捕捉互连线的影响，需要进行精确的电路级模拟。

最为著名的电路模拟器Spice[1]是由加州大学Berkeley 分校在20世纪70年代开发的一个开源软件，它能非常准确有效地模拟电路，但是计算效率比较低下，特别是对于大规模集成电路，进行仿真所需要的时间是无法让人忍受的。虽然之后在Spice的基础上，有Avant公司（现已被synopsys公司收购）逐渐开发出了商业化仿真软件HSPICE及Cadence Design System公司开发出了类似功能的PSPICE，它们相对于Spice的精度和效率均有提高，但是在当今深亚微米工艺尺寸下，这些商业软件仍然不能对于较大规模的电路进行高效率地仿真分析。所以现在工业界急需一个能够在不牺牲很大精度情况下能够快速模拟大规模集成电路的新的仿真器，这也就是模型降阶算法的产生和逐步受到关注的原因。

当前除了以SPICE为代表的基于数值分析的模拟电路仿真器，还有另一大流派即基于符号化分析的模拟电路仿真器。对于数值型电路仿真器，一般情况下，当电路参数发生改变时，需要通过重新执行来获得新的计算结果，因此数值型的电路仿真器虽然可以很好地验证电路设计，但很难预计电路参数改变后的性能情况。灵敏度分析法可以比较好的用于提高模拟电路的性能，这种方法主要是归一化计算电路元件参数对电路性能产生的影响，从而帮助设计者决定如何修改电路参数以获得理性的性能。但是对于一个电路而言计算所有元件的灵敏度往往是没有必要的，而且几乎不可能通过数值计算的方法来计算电路中所有元件之间的灵敏度问题。符号化的分析方法可以比较好地解决这个问题，电路级的符号化分析方法是一种通过电路自变量（时间与频率）、应变量（电压和电流）以及符号化的电路元件来形式化计算电路特性和行为的方法。符号化分析法本身具有很多简化模拟电路设计的特性，因此我们希望在进行模型降阶简化电路的时候可以

将符号化分析考虑进去，对电路进一步简化，降低分析复杂度，保证分析准确性的同时，提高模拟电路分析的效率。

## 1.2 互连线

### 1.2.1 发展趋势

自从集成电路发明以来，芯片已无可辩驳地成为电子电路集成的最终形式。从那以后，集成度增加的速度就按照摩尔定律的预测稳步前进。摩尔定律的预测在未来若干年依然有效的观点目前仍被普遍接受，然而，一个同样被广泛认同的观点是，物理定律将使摩尔定律最初描述的发展趋势停止。在这种情况下，电子电路技术和电路设计的概念将进入一个新的发展阶段，互连线将在重要性和价值方面都得到提升。在被称作“超越摩尔定律”的新兴范式下，无论物理上还是使用上，在 z 轴方向组装都将变得越来越重要。目前在电子工业中第三维正被广泛关注，成为互连技术的主导。

### 1.2.2 互连线模型

#### 1.2.2.1 RC 树模型

大多数互连线模型的拓扑结构都和图 1-1 的树状结构类似。对于这样的电路结构，能包含其所有电路信息的最简单的拓扑结构是 RC 树结构，如图 1-2 所示。

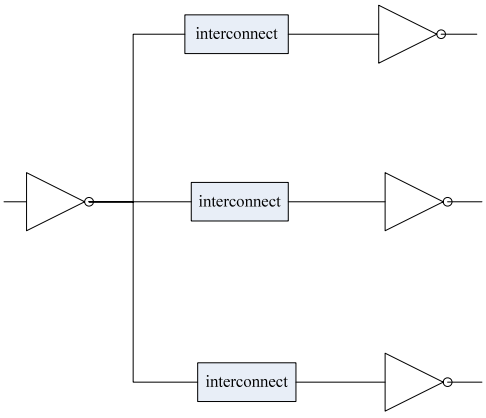



图 1-1 互连线模型

RC 树指的一个 RC 电路，它的每个结点上都有一个电容接地，而且在任意两个非地结点之间没有电容，也没有电阻接地。为了建立这个简单的模型，一般将驱动设定为戴

维宁等效电压源，而接收端由线性电容替代。

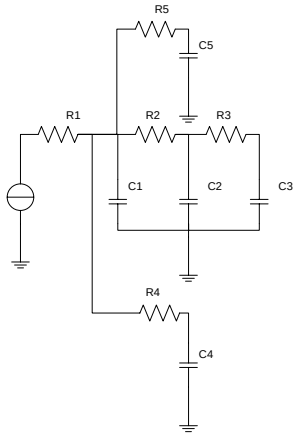



图 1-2 互连线的电子模型

1.2.2.2 RLC 树模型

在上面提到的 RC 树模型中，所有邻近的导体都被模型成地。这个假设只有当串扰效果忽略不计时才是合理的。这个模型告诉我们在互连线中电流的高频部分通过电容返回。但是并没有说电流的直流和低频部分的返回路径。这就是当前板上电感建模面临的主要问题，因为电感一般是闭环的，而且在没有精确分析前很难确定它的返回路径。

当能快速的估计的电感的影响，比如延时等，RLC 树模型在互连线中建模中就会起到很大的作用。通过一些可行的关系返回路径的假设，RLC 树的建模也变的可行。一旦返回路径确定了，就可以通过部分信号的自感来直接计算环路的电感。在统一电流分布的假设下，存在一个闭合的表达式来表达特定几何形状的部分自感电感。

## 1.3 模型降阶

### 1.3.1 简介

针对不同线性电路的特点，人们通常对它们的降阶模型感兴趣，提出了许多模型降阶方法。目前的研究基本上分成两个方向向前发展。

第一类方法是基于矩匹配（Moment Matching）的降阶模型方法。它又分为两类：一类是利用Pade 近似来进行瞬态匹配降阶[2][3][5][6][7][8][10][26][27]。由L. T. Pillage首先提出了著名的AWE(Asymptotic Waveform Evaluation)[3]算法，该算法相对于传统的Spice仿真器来说是一个进步，能够很好地匹配2n阶矩，计算的精确性也能

够得到保障，但是该算法的计算速度还是不够理想。之后，在该算法的基础上，研究者又提出了很多新的算法，这些算法有一个共性，都是建立在由Krylov子空间投影算法产生的降阶模型，避免了AWE算法在距匹配过程中需要直接计算矩的缺点。R. Freund提出了基于Lanczos迭代的Krylov字空间算法而进行Pade近似的降阶算法PVL［5］，之后他又加以改进提出了多端口的MPVL算法［2］，以及对于与RLC电路的单端口的SyPVL［26］及多端口的SyMPVL［27］。另一种基于Krylov子空间的是Anoldi算法，由于它在模型降阶中的应用，相应产生了由A.Odabasioglu提出的PRIMA算法[8]，该算法能够保持电路的稳定性和无源性，但是仅仅针对固定的电路形式。

另一类计算降阶模型的方法是平衡截断实现法（Truncated Balanced Realization）[16]，它是在控制领域的研究中产生的，它能产生一个接近优化的Hankel-norm 近似，并且有著名的L∞ 传输函数误差限。但是它需要求解两个Lyapunov 方程，计算量非常大。Jing-Rebecca Li 等[17]根据这个原理，利用系统的主可控子空间来计算互连线的降阶模型，取得了比较好的效果。最近，Janet M. Wang 等[18]提出了一种基于时域技术的降阶模型方法，对于处理有强烈互感的互连线有非常好的效果。

另外，在经历了对于模型降阶数十年的研究后，现在的研究者已不满足于仅仅进行理想情况下的研究，而是与电子工业的发展相结合起来。由于工艺尺寸的不断缩小，以及在加工过程中各类环境，材料因素的影响，现在的集成电路的产品的性能可能与设计时理想的性能相差很大，而且随着工艺尺寸的继续缩小，这种影响的效果将会更显著。所以我们在模型降阶的过程中也要考虑到电路参数的变化情况对于整个系统的影响。对此，如何在电路的参数变化的情况下进行模型降阶成为了现今的一个研究的热点。Ying Liu[29]分析了参数扰动对于 PACT[23]及 PRIMA 算法的影响，James D.Ma[28]提出了使用有限区间来对参数变化的概率密度函数进行分析，并且与传统的 Monte Carlo 方法结合，并且将他所提出的算法应用到了 AWE 算法和 PRIMA 算法之上进行了误差分析。

### 1.3.2 渐进波形求值法

Lawrence T. Pillage 是把模型降阶算法引入到电路仿真中的先驱，他在1990 年提出的Asymptotic Waveform Evaluation (AWE) 方法使得当时对互连线时延的计算的速度大大提高，这个方法在EDA工具中也被应用。

首先，采用MNA(Modified Nodal Analysis)电路方程公式表示法，一个集总的、线性的、时间为变量的电路通常可以用一阶微分方程表示：

$$C\dot{x} = -Gx + bu \tag{1-1}$$

$$y = l^T x + du \tag{1-2}$$

其中向量x代表电路的变量。矩阵G代表非储能元件的影响，如电阻。矩阵C代表储能元件的影响，如电容和电感。y是激励的输出，bu及du代表来自独立源的激励。传统仿真器spice的模拟过程就是求解以上微分方程以得到电路的各种特性分析数据。所以如何快速高效解该矩阵微分方程成了模型降阶的课题。

AWE 的基本过程是用电路传输函数H(s)的极点和余式的一个子集去近似原来的H(s)：

$$\hat{H}(s)=\frac{k_1}{s-p_1}+\frac{k_2}{s-p_2}+\cdots+\frac{k_q}{s-p_q} \tag{1-3}$$

其中pi 和ki 是分别是极点和余式。q是近似的阶数。这个式子很容易写成时域对应的响应：

$$\hat{h}(t)=k_1e^{p_1t}+k_2e^{p_2t}+\cdots+k_qe^{p_qt} \tag{1-4}$$

这个函数就是我们想得到的对于原系统传输函数的一个近似。

AWE 使用矩匹配（moment-matching）来唯一确定k和p。$\hat{H}(s)$的矩必须保持和实际电路的矩相等。通常我们定义时域函数h(t)的i阶矩为：

$$m_i=\frac{(-1)^i}{i!}\int_0^\infty t^i h(t)dt \tag{1-5}$$

把这个式子应用到$\hat{h}(t)$上我们就得到：

$$\hat{m}_i=\frac{k_1}{p_1^{i+1}}+\frac{k_2}{p_2^{i+1}}+\cdots+\frac{k_q}{p_q^{i+1}} \tag{1-6}$$

所以，对于q 阶的近似会有q个未知极点（p1...pq）和q 个未知余式（k1...kq），一共是2q个未知数。解2q 个未知数需要2q个独立方程，它们是：

$$\begin{aligned}
&k_1+k_2+\cdots+k_q=m_{-1}\\
&\frac{k_1}{p_1}+\frac{k_2}{p_2}+\cdots+\frac{k_q}{p_q}=m_0\\
&\frac{k_1}{p_1^2}+\frac{k_2}{p_2^2}+\cdots+\frac{k_q}{p_q^2}=m_1\\
&\qquad\vdots\\
&\frac{k_1}{p_1^{2q-1}}+\frac{k_2}{p_2^{2q-1}}+\cdots+\frac{k_q}{p_q^{2q-1}}=m_{2q-2}
\end{aligned} \tag{1-7}$$

注意到等式右边是实际电路对于某个特定激励响应的2q个矩。接下去可以使用

Newton-Raphson算法或者其它迭代算法来求解上述方程组。Pillage提出的算法是将电路的矩所满足的关系写成如下矩阵：

$$\begin{bmatrix} m_{-1} & m_0 & \cdots & m_{q-2} \\ m_0 & m_1 & \cdots & m_{q-1} \\ \vdots & \vdots & & \vdots \\ m_{q-2} & m_{q-1} & \cdots & m_{2q-3} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{q-1} \end{bmatrix} = \begin{bmatrix} m_{q-1} \\ m_q \\ \vdots \\ m_{2q-2} \end{bmatrix} \tag{1-8}$$

而由该矩阵求出的解 $a_i(0 \le i \le q-1)$ 与 $p_i(1 \le i \le q)$ 满足， $p_i$ 是以 $a_i$ 为系数的下述方程的解：

$$a_0 + a_1 p^{-1} + a_2 p^{-2} + \cdots + a_{q-1} p^{-q+1} + p^{-q} = 0 \tag{1-9}$$

这样在求出 $p_i$ 后代入到(1-7)式中继续求出 $k_i(1 \le i \le q)$ 的值，这样便求出了近似传递函数(1-3)式中所有的未知量。可以看出AWE算法在求解的过程中需要解线形方程组(1-8)，需要求解高次方程的根(1-9)，需要根据(1-5)式求出电路的2q阶矩，实际的计算量比求解最初的传输函数H(s)减少了很多但也是相当大的，所以由于AWE算法的这些缺陷，导致了后来PRIMA，PVL，SPRIM等算法的产生。

### 1.3.3 RICE 算法

上面讨论了AWE算法的过程,我们不难发现该算法一个重要的条件就是要已知电路的2q阶矩的值，但是对于一个给定的电路，这些矩的值也是需要我们计算得到的，所以为了快速求解这些矩的值Pillage及Ratzlaff提出了RICE(Rapid Interconnect Circuit Evaluation)算法。下面简单的介绍RICE算法的流程。

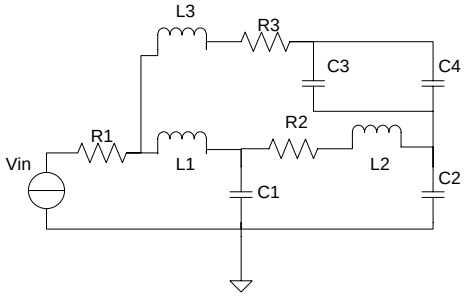



图1-3 基本RLC电路

图1-3所示为一个基本的RLC电路，它包含电源、电阻、电容和电感器件，假设在0时刻该电路在电源的作用下处于稳态，则此时刻我们记下通过各电容的电流和各电感两

端的电压作为电容和电感的0阶矩，如下图1-4所示。

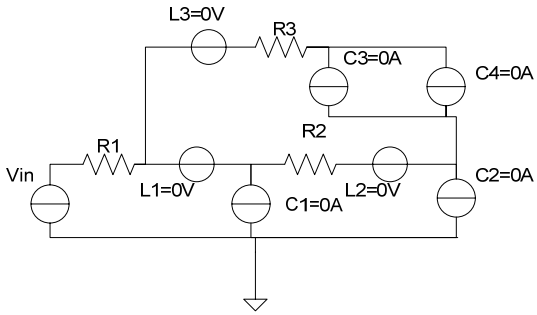



图1-4 RICE算法0阶矩求解

接下去我们将电路的电压源置0，然后将每一个电感看成是一个电压源，它的数值大小是该电感的大小乘以上一步求出的该电感两端的电压，同时将每个电容看成是一个电流源，它的数值大小是该电容的大小乘以上一步求出的通过该电容的电流大小，之后继续对于变换的电流作DC分析，而在这一步求出的每个电容所代表的电流源两端的电压和通过每个电感所代表的电压源的电流作为电感和电容的0阶矩。

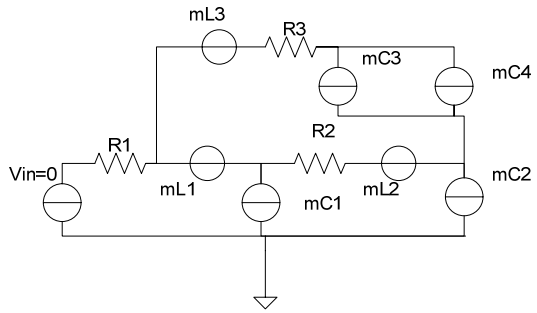



图1-5 RICE算法1阶矩求解

以上是 RICE 算法思想的一个介绍，在实际求解过程中将电路作为生成树进行求解，图 1-6 是图 1-3 所示电路的一个基于树的表述形式，RICE 中使用了虚拟遍历的算法减少了为了求解电路所有的矩所要求的遍历树的次数。

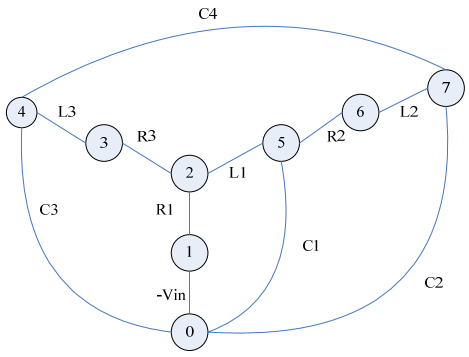



图1-6 图1-3所示电路的生成树表示

如上图所示，首先对于图中的每个结点，由叶结点开始进行一次反向深度优先遍历，这样的遍历可以保证每个结点只有在所有支路电流都知道的情况下才会被访问，因此在经过一次反向深度优先遍历之后，就可以知道所有支路上的电流。当求出所有电流之后，从根节点（图中即 0 点）对由 R 和 L 组成的 tree-branch 进行一次深度优先遍历，这样的遍历可以保证所有结点都是在已知前一个结点电压的情况下才会被访问，因此在一次深度优先遍历之后就可以知道所有结点电压。因此，在两次遍历之后，可以得出 L 和 C 在这一阶的矩量。

RICE 算法的应用大大提高了 AWE 算法的效率，以至于后续的 PRIMA 算法也是建立在 RICE 算法求解矩的基础上进行的。

## 1.4 本章小结

本章主要简单介绍了互连线发展的趋势以及当前研究的热点之一，模型降阶问题。对于模型降阶问题，通过简单的介绍 L.T.Pillage 提出的经典的 AWE 算法和其衍生的 RICE 算法，使得我们对于模型降阶中出现的电路矩有了初步的了解。在下面的章节中，就电路矩和 Elmore 延时之间的关系，以及电路矩的计算过程，RICE 算法的理论根据有更详细的分析。

# 第二章 Elmore 延时和矩

对于 IC 互连线，最简单的预估其性能的方法是用电路本身的延时进行计算。因此，这些年来，对于 RC 树状电路延时的估计十分的热门。很多研究都是基于电路的延时。这一章，将具体分析下电路延时中最著名、对于研究影响也最大的一阶延时——Elmore 延时。

## 2.1 传输函数的延时

对于大多数 RC 互连线电路，都是通过输入端的激励和输出端的电压值来衡量一个电路的特性。而这两者之间的关系正好就是传输函数 $H(s)$，下图 2.1 是一个简单的线性电路框图。

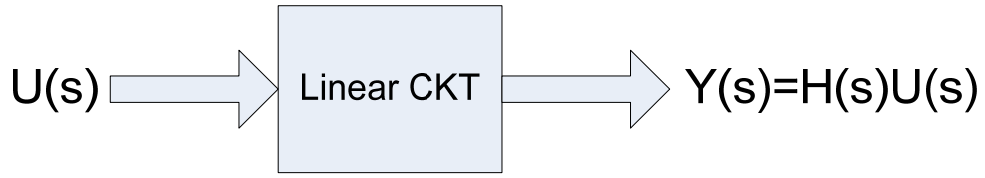



图 2-1 简单单输入单输出线性电路

### 2.1.1 电路的传输函数和导抗函数

当初始条件为零时，我们可以定义一个传输函数。如上图 2-1 所示的简单单输入单输出电路，它的传输函数就是输出和输入的比值：

$$H(s)=\frac{Y(s)}{U(s)} \tag{2-1}$$

导抗函数是传输函数的重要组成部分，当传输函数在工作点时就是导抗函数，它有两种形式。第一，如果输入是电流源，输出是以电压的方式衡量的，那么此时的导抗函数就是输入阻抗。第二，如果输入是电压源，而输出是以电流的方式衡量的，那么此时的导抗函数就是输入导纳。

通常情况下，习惯将传输函数在 s 域上表示成两个多项式的比值：

$$H(s)=\frac{b_0+b_1s+\bullet\bullet\bullet+b_ms^m}{1+a_1s+\bullet\bullet\bullet+a_ns^n} \tag{2-2}$$

在上式(2-2)中，所有的系数 $a_i$（$i=1,2,...,n$）和 $b_j$（$j=1,2,...,m$）都是实数。通过将公式(2-2)中因式分解，我们可以得到一个全新的传输函数表达式：

$$H(s)=K\frac{(s-z_1)(s-z_2)...(s-z_m)}{(s-p_1)(s-p_2)...(s-p_n)} \tag{2-3}$$

其中，K 是系数，应当等于 $\frac{b_m}{a_n}$，而 $p_i$（$i=1,2,...,n$）就是该电路所具有的极点，相应的， $z_j$（$j=1,2,...,m$）就是该电路所具有的零点。极点和零点有可能以共厄复数的形式出现。不考虑 K 的因素，一个传输函数完全可以在复数平面通过极点和零点表示出来。

尽管传输函数的零极点表示法对于模拟电路的设计和分析非常有用和方便，但在互连线时域分析中更多采用的还是极点留数表示法。假设传输函数中的极点都是唯一的，那么传输函数通过极点留数表示法可以写成：

$$H(s)=d+\sum_{i=1}^{n}\frac{k_i}{1-p_i} \tag{2-4}$$

### 2.1.2 时域的响应和延迟

在线性电路中，可以通过 Laplace 变换和逆变换得到时域和 s 域上电路的相应关系。假设 $h(t)$ 是 $H(s)$ 相对应的时域响应，那么我们可以得到这两者之间的关系如下表示：

$$H(s)=\int_0^{\infty}e^{-st}h(t)dt \tag{2-5}$$

当输入值为单位值，如果传输函数 $H(s)$ 是复杂的频域输出，那么时域响应 $h(t)$ 就是一个冲激输入的响应，即冲激相响应。如果可以通过如式(2-4)的极点留数表达形式表示出这个传输函数，那么该冲激响应可以写成如下形式：

$$h(t)=d\delta(t)+\sum_{i=1}^{n}k_ie^{p_it}u(t) \tag{2-6}$$

其中，$\delta(t)$ 为单位冲激函数，而 $u(t)$ 为单位阶跃函数。

对于一个单位阶跃输入，通过式(2-1)，可以得到其在 s 域上的输出端表达式：

$$Y(s) = H(s)\frac{1}{s} \tag{2-7}$$

用极点留数表示法替换式（2-7），得到如下：

$$Y(s) = \frac{d}{s} + \sum_{i=1}^{n} \frac{k_i}{(s - p_i)s} \tag{2-8}$$

对上式进行部分分式分解，得到新的公式：

$$Y(s) = \frac{d}{s} + \sum_{i=1}^{n} \frac{k_i}{p_i}\left(\frac{1}{s - p_i} - \frac{1}{s}\right) \tag{2-9}$$

此时 $Y(s)$ 在时域上的响应可以写成如下形式：

$$y(t) = (d + \sum_{i=1}^{n} \frac{k_i}{p_i}(e^{p_i t} - 1))u(t) \tag{2-10}$$

这种通过 Laplace 逆变换得到时域响应表达式的方法可以在任何输入源为线性波形的电路中应用，因为所有的线性输入源都可以拆分成数个不同的时移斜坡函数。

电路的延时可以通过解式(2-10)得到，例如对于一个单调响应，当输入为阶跃输入信号，假设稳定状态的响应也是单位的，那么这个单调响应的 $\alpha$% 的延时可以通过求解 $\tau$ 得到：

$$d + \sum_{i=1}^{n} \frac{k_i}{p_i}(e^{p_i \tau} - 1) = \alpha \tag{2-11}$$

在计算延时 $\tau$ 的过程中，会遇到 2 类问题，其一就是如何获得极点和留数。对于大多数互连线电路，几乎不可能通过计算得到所有的极点和留数，一般只能获得该电路的主极点及其对应的留数。另一个问题就是对于某些电路，比如含有环路的电路，在计算延时的过程中将花费大量的时间。

## 2.2 RC 互连线的延时

不论是在时域上还是在频域上对互连线进行分析，通常采用的模型都是 RC 树状模

型。对于如图 2-2 所示的门级互连线电路，使用最为广泛的模型就是 RC 树状模型，如图 2-3 所示。对于一个 RC 树，它是一个完全由 R 和 C 构成的电路，其中，所有的电容 C 都连接在每个结点和地之间，在 2 个非地结点中不会存在电容，也不会有电阻连接到地。

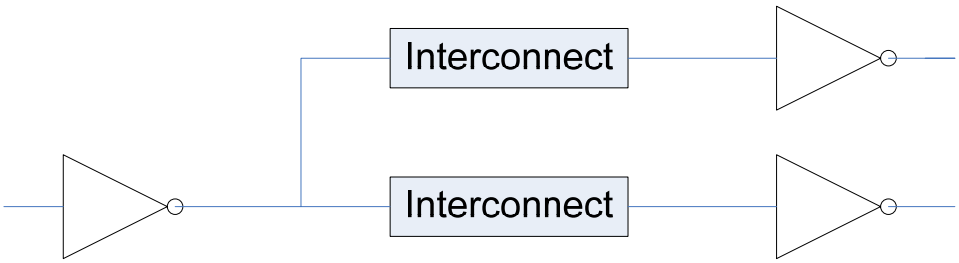



图 2-2 门级互连线电路

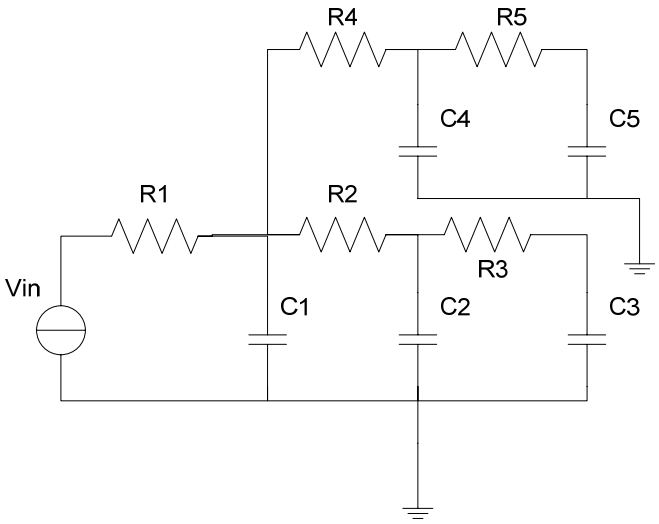



图 2-3 简单 RC 树电路

通过将门级互连线电路转换成 RC 树状模型，对于互连线延迟就可以通过近似求解谱的方法估计得到。Elmore 延时，就是冲激响应的一阶矩表达式，由于其简单所以被应用的最为广泛。Elmore 延时的最大优点就是它是由 RC 参数值确定的既简单且闭合的表达式。但是对于深亚微米级技术生成的电路，Elmore 延时就会体现出它的局限性，仅能提供有限的效率。

## 2.3 Elmore 延时

### 2.3.1 Elmore 延时简介

Elmore 延时最早于 1948 年提出，当时提出 Elmore 延时是为了估计放大器电路的延时。后来，由于它的阶跃响应的特性，Elmore 延时被用来估计 RC 树的延时。比如说，对于图 2-3 中$C_3$的结点电压的阶跃响应如下图 2-4 所示，在图 2-4 中还显示了相同结点的单位冲激响应。

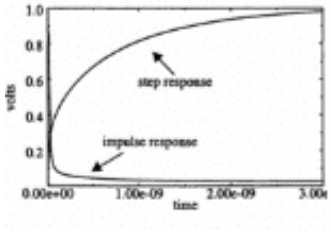



图 2-4 $C_3$的结点电压的阶跃响应和单位冲激响应

由于阶跃响应是冲激响应的积分形式，因此对于单调的阶跃函数，50%的延时点正好是冲激响应函数面积一半所在的时间，即：

$$\int_0^{\tau} h(t)dt = 0.5 \tag{2-12}$$

在图 2-5 中，可以发现 Elmore 延时通过非负的冲激响应函数$h(t)$的均值来近似时间$\tau$，一个单调阶跃函数的延时。如果将图 2-5 中的非负冲激响应看作一个概率密度函数，那么它的均值就可以由冲激响应的一阶矩来近似得到。因此，此时的 Elmore 单位阶跃响应延时的近似$T_D$可以表示成：

$$T_D = \int_0^{\infty} th(t)dt \tag{2-13}$$

而此时，在$h(t)$下的面积为单位值，即

$$\int_0^{\infty} h(t)dt = 1 \tag{2-14}$$

对于这种近似，观察图 2-5 可以发现，当冲激响应为对称函数时，此时的均值正好对于它的中值，然而当图 2-4 中的情况，即冲激响应为非对称函数时，均值就不再正好是中值，也就是 Elmore 延时并不能准确的表达实际的延时。

### 2.3.2 计算 Elmore 延时

Elmore 延时对于 RC 树是一种简单方便的近似，因为 Elmore 延时对于这种具有特殊拓扑结构的电路，可以十分简单的计算得到。总的来说，通过有效的路径追踪的算法，可以通过线性次遍历树就能计算得到 Elmore 延时，在结点 $i$ 的 Elmore 延时可以表示成如下形式：

$$T_{D_i} = \sum_{i=1}^{N} R_{ki} C_k \tag{2-15}$$

其中，$R_{ki}$ 代表电路中电阻的部分和，包括从输入端至结点 $i$ 中（这种情况仅为电路只有一条支路），$C_k$ 代表的是结点 $k$ 与地端之间的电容。例如，对于图 2-3，$C_3$ 两端的 Elmore 延时具有如下的表达式：

$$T_{D_3} = R_1C_1 + (R_1 + R_2)C_2 + (R_1 + R_2 + R_3)C_3 + R_4C_1 + R_5C_1$$

## 2.4 矩的概念

通过前面的介绍，冲激响应的一阶矩定义了 Elmore 延时，但只有通过高阶矩，才能更好的认识到这种近似的局限性。为了更方便的理解高阶矩，考虑将在 Laplace 域上的传输函数写成如下形式：

$$H(s) = \frac{b_0 + b_1 s + \bullet\bullet\bullet + b_m s^m}{1 + a_1 s + \bullet\bullet\bullet + a_n s^n} \tag{2-16}$$

将上述传输函数在 $s = 0$ 处展开，可以得到新的传输函数：

$$H(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \ldots \tag{2-17}$$

其中，

$$m_q = \frac{1}{q!}\frac{d^q H(s)}{ds^q}\Big|_{s=0} \tag{2-18}$$

时域和频域之间的关系仍然遵循 Laplace 变换的关系：

$$H(s) = \int_0^\infty h(t)e^{-st}dt \tag{2-19}$$

将上式(2-19)中的$e^{-st}$在$s=0$处展开，得到下式：

$$\begin{aligned} H(s) &= \int_0^\infty h(t)(1 - st + \frac{1}{2}s^2t^2 - \frac{1}{6}s^3t^3 + \cdots)dt \\ &= \sum_{k=0}^{\infty}\frac{(-1)^k}{k!}s^k\int_0^\infty t^k h(t)dt \end{aligned} \tag{2-20}$$

从式(2-20)中，可以得到冲激响应$h(t)$的第$q$个系数是

$$m_q = \frac{(-1)^q}{q!}\int_0^\infty t^q h(t)dt \tag{2-21}$$

而在概率分布理论中，对于函数$h(t)$的第$q$阶矩的定义如下：

$$M_q = \int_0^\infty t^q h(t)dt \tag{2-22}$$

可以发现，这两者之间是有区别的，两者之间的关系为

$$m_q = M_q\frac{(-1)^q}{q!} \tag{2-23}$$

对于这两个矩，$m_q$为电路矩，而$M_q$为概率矩。

### 2.4.1 计算矩

当计算得到电路的矩之后，它对于分析电路的特性是十分有帮助的，因此如何有效的计算矩就现的十分重要。下面将介绍一种简单的求解 RC 树状电路各阶矩的方法，RC 树状电路如下图 2-5 所示。

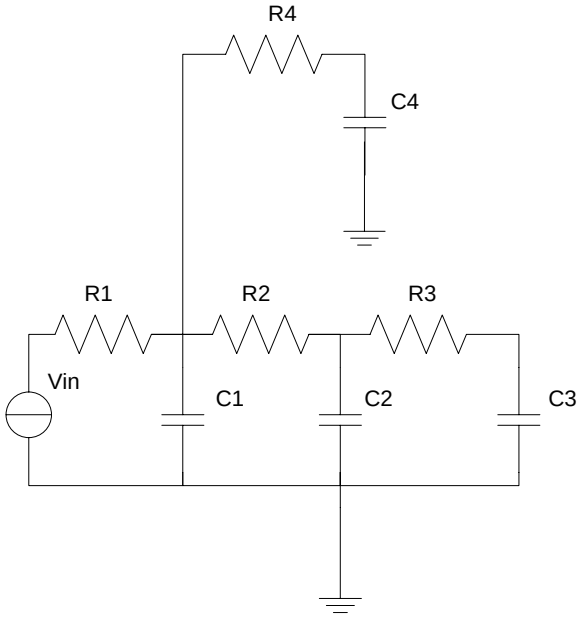



图 2-5 简单的 RC 树状电路

上图 2-5 中的 RC 树状电路在 Laplace 域可以用如下图 2-6 中的展开式表示，其中电容用复杂导纳所取代。假设所有的电容两端的电压都可以表示称在 s 域上无限项的和。

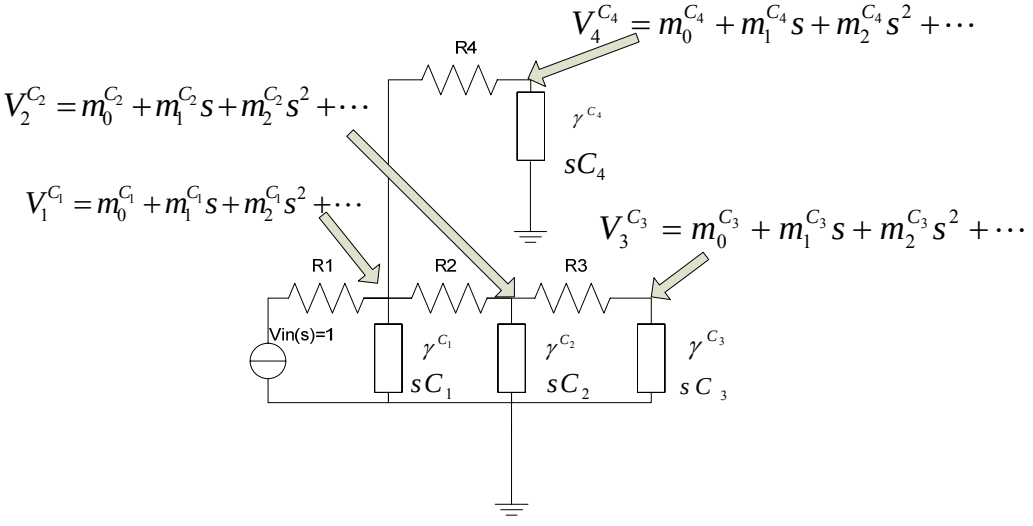



图 2-6　图 2-5 的简单 RC 树电路在 s 域上的表示

其中，$m_j^{C_i}$ 表示不同结点处的不同阶的矩。将电容两端电压用此方式表达而且知道电容值之后，我们可以用同样的方法写出流经电容的电流值。这样，知道流经电容的电流之后，就可以用这些电流来替代导纳，如下图 2-7 所示。在图 2-7 中，仅有 $m_j^{C_i}$ 是还未知的项。

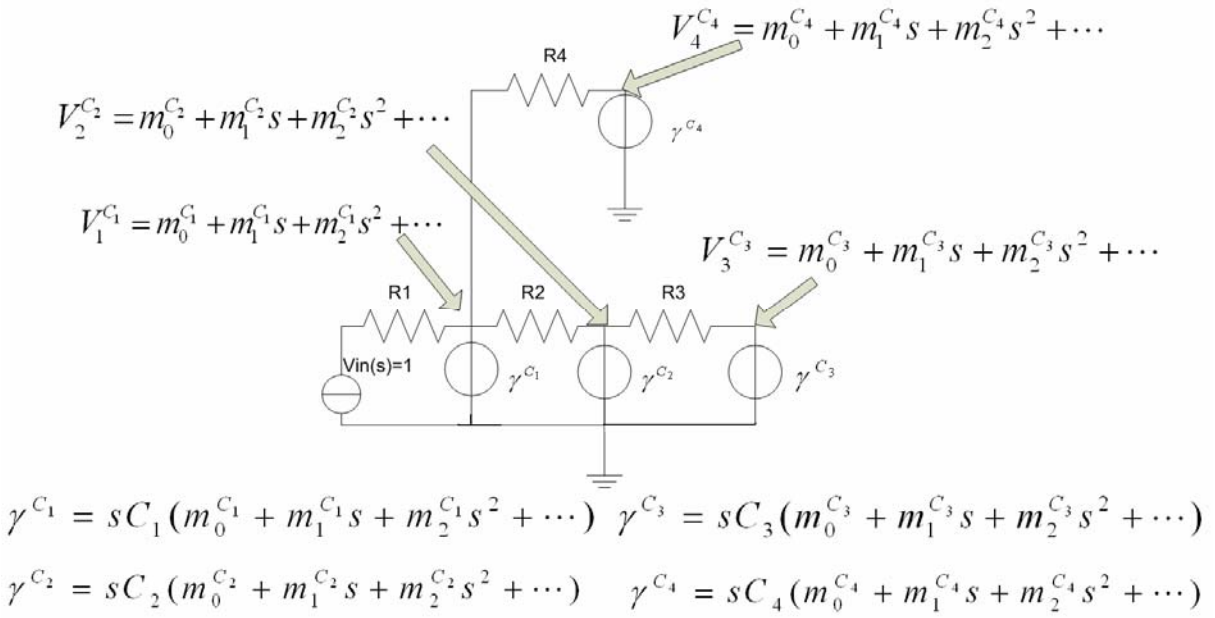



图 2-7　图 2-6 的等效电路及其各点电压解

在图 2-7 中，要求解所有电容电压的 $m_0$ 项，可以通过设定 $s=0$，由于在 s 域上没有常数项，将图 2-7 中的电流源设为 0，再利用图 2-7 中的公式求解直流分析。对于上述电路图，明显的，所有电容的 $m_0$ 项都等于 1。

这里将电容初始值设为 0 来计算电路各点的 $m_0$ 项，对于所有的电路拓扑结构都是成立的，如图 2-8 所示。相同的，如果考虑电感 L，电路变为 RLC 树状电路，那么此时计算电路各点的 $m_0$ 项时，电感 L 将会被值为 0 的理想电压源所替代，而电容仍旧被值为 0 的理想电流源所替代。

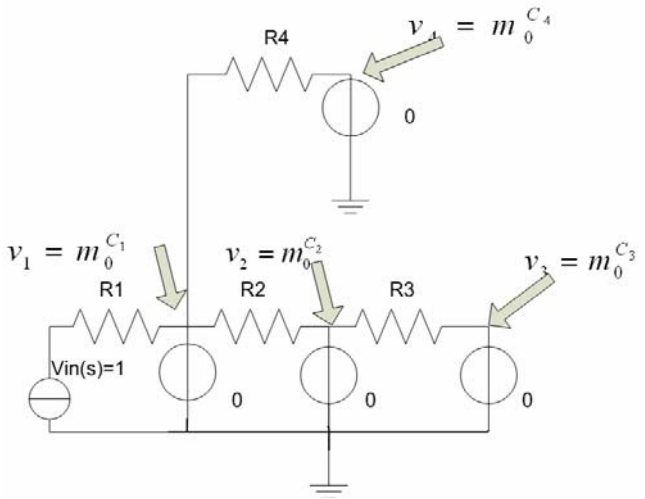



图 2-8 图 2-5 RC 树电路的求解 $m_0$ 时的等效电路图

下面求解 $s$ 前面的系数 $m_1$，考虑图 2-7，观察电流和电压的公式，我们可以发现，电压表达式中的 $s$ 项是由电流源中的 $s$ 项产生的。由于在上轮求解 $m_0$ 项的过程中，我们已经求得了电流表达式中的 $s$ 项。因此，我们可以通过将所有的电容设定为值为 $C_i m_0^i$，然后求解各点的直流电压来求解 $m_1$ 项。此时，直流电压源在 $s$ 前的系数为 0，因此它对于计算 $m_1$ 项不会产生任何影响。如图 2-8 所示，所有元件通过这样设定之后，通过解直流方程就能求得所有的 $m_1$ 项。

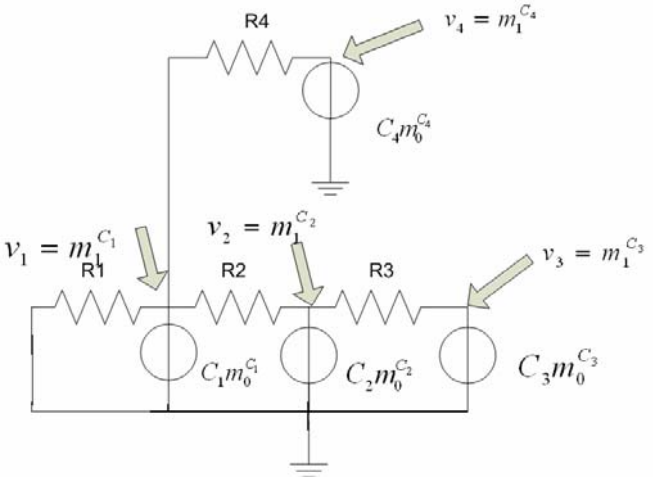



图 2-9 图 2-5RC 树电路的求解 $m_1$ 时的等效电路图

利用相同的原理，我们就可以轻松的求得所有结点更高阶的矩，如下图 2-10 所示。

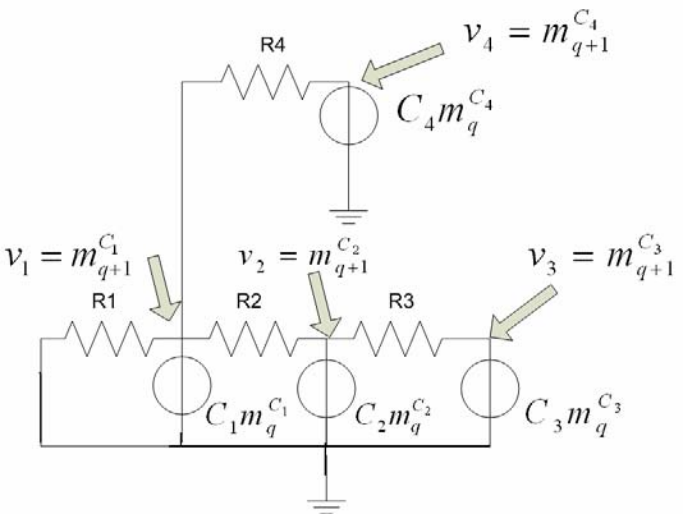



图 2-10 图 2-5 的 RC 树电路求解 q+1 阶矩时的等效电路

### 2.4.2 中心矩

Elmore 延时的精确度通常由冲激响应均值和中位值所分布的地方决定，一般没有确定的方法来判断 Elmore 延时的误差有多少，但是可以通过高阶矩的信息来判断它的相对精确度。可以通过中心矩来刻画高阶波形。

和矩一样，中心矩也来自于概率分布理论。通过分析 Elmore 函数的分布，我们可以用它来近似 Elmore 延时。考虑上节提到过的电路矩的表达式：

$$m_q = \frac{(-1)^q}{q!}\int_0^\infty t^q h(t)dt \tag{2-24}$$

而冲激响应的均值是

$$\mu = \frac{\int_0^\infty th(t)dt}{\int_0^\infty h(t)dt} = \frac{-m_1}{m_0} \tag{2-25}$$

当系统的直流增益为单位时，有 $m_0 = 1$， $\mu = -m_1$

冲激响应的中心矩是关于均值的矩，它的定义如下：

$$\mu_i = \int_0^{\infty} (t-\mu)^i h(t)dt \tag{2-26}$$

很简单的，通过公式迭代计算，可以得到前面的几阶中心矩：

$$\mu_0 = m_0 \tag{2-27}$$

$$\mu_1 = 0 \tag{2-28}$$

$$\mu_2 = 2m_2 - \frac{m_1^2}{m_0} \tag{2-29}$$

$$\mu_3 = -6m_3 + \frac{6m_1m_2}{m_0} - 2\frac{m_1^3}{m_0^2} \tag{2-30}$$

和冲激响应的矩不同的是，中心矩有几何意义：

$\mu_0$ 是曲线下所覆盖的面积，一般性为单位值，或者就是简单的尺度变换下；

$\mu_2$ 是用来衡量曲线至中心的分布的方差。一个较大的方差意味着较为广泛的分布；

$\mu_3$ 用来衡量曲线分布的偏移度。

## 2.5 本章小结

Elmore 延时对于分析时序特性非常的重要，几乎在各个层面的电子电路设计自动化中都会有所应用。对于 RC 树电路，阶跃响应的 50％处的延时一般都有一个上界，也意味着输入信号有个有限的上升时间。在电路的模型降阶领域中，最为关键的地方就在于求解电路矩的方法。一个有效的求得电率矩的方法会使得电路分析的效率大为提高，本章简单介绍了矩的求解过程。下一章，会对已有的求解方法进行新的探索和分析。

# 第三章 符号化分析

## 3.1 符号化理论

电路的符号化分析历史悠久。上世纪 60 年代末及 70 年代初期，随着计算机技术的进步，符号化电路分析方法成为研究的热点，其代表有 SNAP[4]和 NAPP[9]，他们主要用来分析模拟滤波器。基于图形的分析方法（例如生成树枚举法和符号流图法）被认为是最适合用于分析小规模整体均符号化的电路；若只将频率看作符号，可以采用数值的方法来分析规模较大的电路；将上述两种方法结合，又诞生了所谓的符号数值混合分析的方法，也就是将小部分的电路参数符号化，从而快速地分析规模较大的电路；随着 SPICE 的诞生，符号化电路分析的优势逐渐被 SPICE 准确和快速的数值结果所掩盖。到了上世纪 80 年代，为了克服符号化分析对电路规模的限制，层次化分解的分析方法被提出，电路符号表达式不再是通过展平整个电路来获得而是嵌套式地产生；同时基于矩阵行列式的分析方法得到发展，它们可以像图形分析方法一样对整体都符号化的电路进行有效的分析。从上世纪 80 年代后期开始，符号化电路分析方法开始焕发新的活力，主要是出于对于集成模拟电路分析的需求，诞生了许多成功的符号化仿真器，如 ISAAC[11][12]，ASAP[13][14]，SYNAP[15][19]，SAPEC[19]，SSPICE[20]，SCYMBAL[21]，SCAPP[22]和 GASCAP[24]。在这些工具中，最灵活和最有效的工具是其中基于行列式的分析法以及符号流图的方法。符号化电路分析方法重获新生主要有两大驱动力，首先是计算机计算性能的大幅提高以及相关高效算法的发展，其次是模拟集成电路设计对计算机辅助设计和设计自动化的迫切需求。

## 3.2 BDD

二叉决策图（BDD）最早由 Lee 等人于 1959 年提出，后来 Akers 于 1978 年将二叉决策图做了进一步的推广，但由于 BDD 并不具有正则性(canonicity)，因此并未引起人们的注意。Byrnat 在 1986 年又对二叉决策图做了深入的研究，对二叉决策图的变量顺序(ordering)进行了限制，并研究了化简方法，从而提出了简化的、有序的二叉决策图(Reduced Ordered Binary Decision Diagram, ROBDD)的概念。由于 ROBDD 具有正则性，

即对于任意一个逻辑函数，它的 ROBDD 表示形式是唯一的，因此 ROBDD 开始得到广泛的研究和应用。绝大多数的逻辑函数都可以用大小合理的 ROBDD 来表示，逻辑函数的运算复杂度和 ROBDD 的大小成正比，即它的复杂性是线性的。此外 ROBDD 与其他的布尔代数表示方法相比，所需的存储空间小，计算速度快，且已有的优化方法均可借助于 ROBDD 获得很高的效率而无需改变算法本身。正是由于 ROBDD 的这些优点，ROBDD 被广泛应用在逻辑综合、测试生成以及形式验证等许多领域。

### 3.2.1 BDD 简介

BDD 算法是一个有向无环的图形，简称 DAG。每个 BDD 的图形有 2 个接地点，0 和 1，分别代表布尔函数 0 和 1。每一个非接地点的结点都有其特定的布尔变量 $v$，而且它都会有 2 个子结点，分别联结到 1（或者 then）和 0（或者 else）。对于每个非地点的结点，它都代表了一个布尔函数，当其变量为 1 时，指向 1（或者 then）这条边，记为正边，当其变量为 0 时，指向 0（或者 else）这条边，记为负边，如下图 3-1 所示，其中实线代表正边，虚线代表负边。

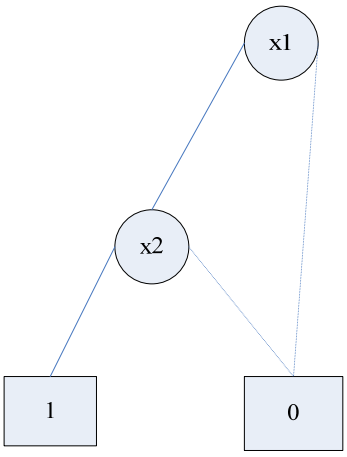



图 3-1 简单 BDD 示意图

OBDD 是 Ordered Binary Decision Diagram 的缩写，是有序的 BDD 算法。因此，对于 OBDD，所有的输入变量都按照固定的顺序排成队列，所有源点至地点间的路径都需要按照输入变量的数许访问。

上文提到的 ROBDD 是在 OBDD 的基础上，每个结点都代表了唯一的逻辑函数，即不会在一个图中出现 2 个或者 2 个以上的结点拥有重复相同的逻辑函数。这就意味着可以减少大量不必要的冗余项。

一个 BDD 与它所表示的布尔代数之间的关系可以按照如下递归定义:

1. 如果 $v$ 节点是接地点，则

(a) 如果 $value(v)=1$，则 $f_v=1$

(b) 如果 $value(v)=0$，则 $f_v=0$

2．如果节点 $v$ 不是接地点且它的 index 为 $i$，则

$$f_v(x_1,x_2,\cdots,x_n)=\overline{x}_i f_{low(v)}(x_1,x_2,\cdots x_{i-1},0,x_{i+1}\cdots,x_n)+x_i f_{high(v)}(x_1,x_2,\cdots x_{i-1},1,x_{i+1}\cdots,x_n) \quad (3\text{-}1)$$

由于 BDD 没有对变量序进行限制，所以任意一个布尔代数可以用许多 BDD 来表示，也就是说，BDD 的表示形式并不是不唯一的。

### 3.2.2 BDD 特性

假设现在有逻辑表达式 $f_1=x_1x_2+x_3x_4+x_5x_6$，按照 $(x_1,x_2,x_3,x_4,x_5,x_6)$ 的顺序建立 BDD 树，如下图 3-2 所示。

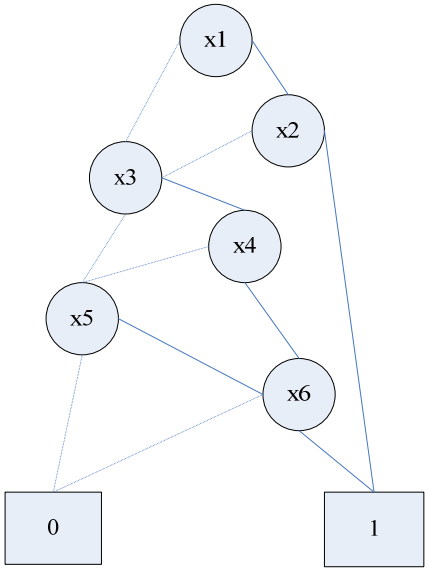



图 3-2 逻辑表达式 $f_1$ 按照 $(x_1,x_2,x_3,x_4,x_5,x_6)$ 的的顺序建立的 BDD 图

这种情况下的 BDD 图十分简单，现按照 $(x_1,x_3,x_5,x_2,x_4,x_6)$ 的顺序建立 BDD 树，如下图 3-3 所示，明显要复杂的多。

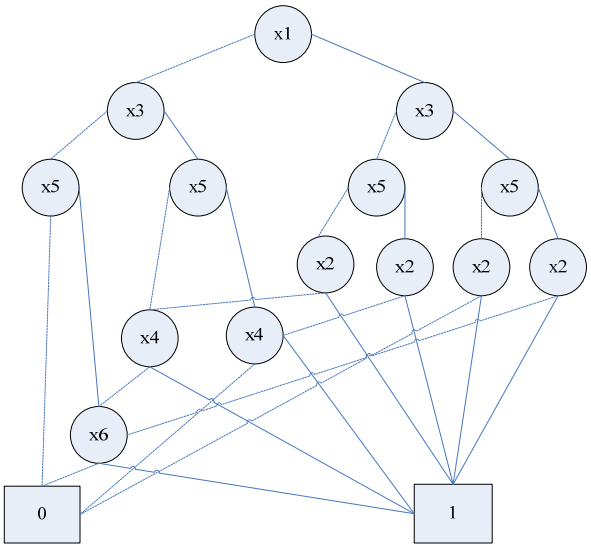



图 3-3　逻辑表达式 $f_1$ 按照 $(x_1, x_3, x_5, x_2, x_4, x_6)$ 的的顺序建立的 BDD 图

可见，对于任意的 BDD，输入变量的排序方式会影响到 BDD 树的结构。这也正是 OBDD 提出的原因。由于在计算机运算中，BDD 中节点数和路径的深度分别对应着计算机实现算法的空间复杂度和时间复杂度，因此很显然，一个好的变量排序方式会影响 BDD 算法的效率，而如何选取最为优化的排序方式也就成为了当前国内外所研究的热点。

现在假设有逻辑表达式 $f_2 = x_1x_2x_3 + x_1x_3 + x_2x_3$，而且已有其最为优化的排序方式，那么可以做出它的 OBDD 图，顺序按照变量 x 的下标升序排列。那么可以得到图 3-4(a)所示。

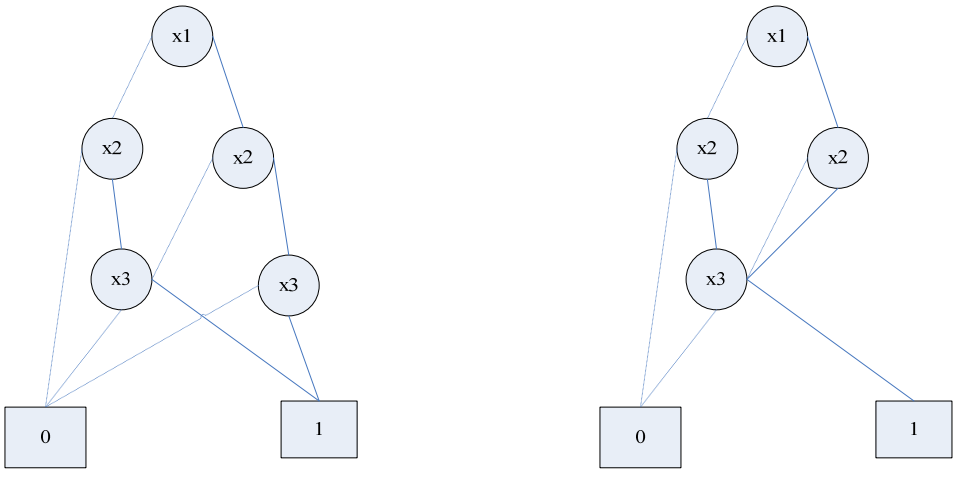



图 3-4 逻辑表达式 $f_2$ 的 OBDD 图

从图中，我们不难看出，变量$x_3$的两个结点具有相同的布尔函数，因此对于这两个结点，其中一个是冗余项，可以去掉，去掉之后的 OBDD 图变为图 3-4(b)。

继续观察图 3-4(b)，发现变量$x_2$的右边结点的边都指向$x_3$这个结点，因此由香农展开式我们可以知道，这个结点也是冗余项，应当删除。

经过删除之后，我们就能得到最后最简形式的 OBDD 图，如图 3-5 所示：

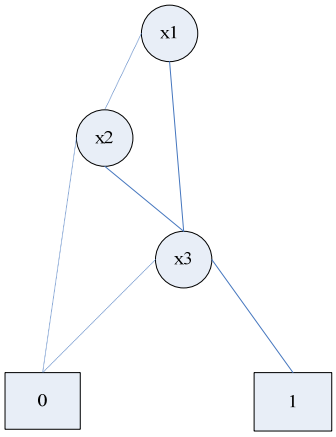



图 3-5 化为最简形式的 OBDD 图

这是再看这个 OBDD 图，它所具有的函数表达式为：$f_2 = x_1x_3 + x_2x_3$，经过验算可知是原来给出的函数表达式的化简形式。

因此，一个 ROBDD 图有如下性质：

（1）对于任意两个同构的 ROBDD 图 G1 和 G2，当且仅当他们的结点集 V1 和 V2 之间存在一一映射的关系。对于任一结点$v_1 \in V1$，必然存在一个结点$v_2 \in V2$，使得$index(v_1) = index(v_2)$，或者当结点为地点时，$value(v_1) = value(v_2)$以及该结点的 2 个子结点都有$low(v_1) = low(v_2)$和$high(v_1) = high(v_2)$

（2）对于 OBDD 中任意一个非源点$v$，节点$v$的子图定义为以$v$为根节点的 OBDD。

（3）对于任意两个同构的 ROBDD 图 G1 和 G2，对于任一结点$v_1 \in V1$，必然存在一个结点$v_2 \in V2$，使得$v_1 = v_2$，而且他们的子图也是同构的。

（4）一个 OBDD 是最简的，当且仅当对于每个节点$v$都存在$low(v) \neq high(v)$，以及任意

两个节点的子图都不是同构的，此时的 OBDD 就是最简有序二又决策图(Reduced Ordered Binary Diagram，ROBDD)。

（5）对于任意一个逻辑函数，f 在给定了它的变量顺序后，存在一个唯一的 ROBDD，它含有最少的节点，其他任何形式的 OBDD 都将包含更多的节点。

ROBDD 的唯一性具有重要的用途。为了判断两个逻辑函数的等价性，只需要转换为两个 ROBDD 的同构问题即可，其时间复杂度与 ROBDD 中所包含的节点个数成正比。

### 3.2.3 ROBDD 的运算

所有的 ROBDD 运算都是基于 ITE，其公式如下：

$$ite(F,G,H)=FG+\bar{F}H \tag{3-2}$$

这个公式定义了一个布尔函数，即如果满足 F，那么就等于 G，否则就等于 H。这个公式和建立 BDD 时所提到的香农展开式形式一样，

$$f=vG+\bar{v}H \tag{3-3}$$

只不过这里的$v$是变量，而 F 是函数。

利用这个 ite 公式，可以表示所有的运算，比如

$$f\bullet g=ite(f,g,0)$$

$$f+g=ite(f,1,g)$$

下表 3-1 列出了所有 16 种可能的运算。

| 逻辑运算 | 表达式 | ITE 等价形式 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| $AND(f,g)$ | $fg$ | $ite(f,g,0)$ |
| $f>g$ | $f\bar{g}$ | $ite(f,\bar{g},0)$ |
| $f$ | $f$ | $f$ |
| $g$ | $g$ | $g$ |
| $f<g$ | $\bar{f}g$ | $ite(f,0,g)$ |
| $XOR(f,g)$ | $f\oplus g$ | $ite(f,\bar{g},g)$ |
| $OR(f,g)$ | $f+g$ | $ite(f,1,g)$ |

| | | |
|---|---|---|
| $NOR(f,g)$ | $\overline{(f+g)}$ | $ite(f,0,\overline{g})$ |
| $XNOR(f,g)$ | $\overline{(f\oplus g)}$ | $ite(f,g,\overline{g})$ |
| $NOT(f)$ | $\overline{f}$ | $ite(f,0,1)$ |
| $NOT(g)$ | $\overline{g}$ | $ite(g,0,1)$ |
| $NAND(f,g)$ | $\overline{fg}$ | $ite(f,\overline{g},1)$ |
| $f\geq g$ | $f+\overline{g}$ | $ite(f,1,\overline{g})$ |
| $f\leq g$ | $\overline{f}+g$ | $ite(f,g,1)$ |

表 3-1 16 种运算的逻辑运算与 ite 等价表示形式

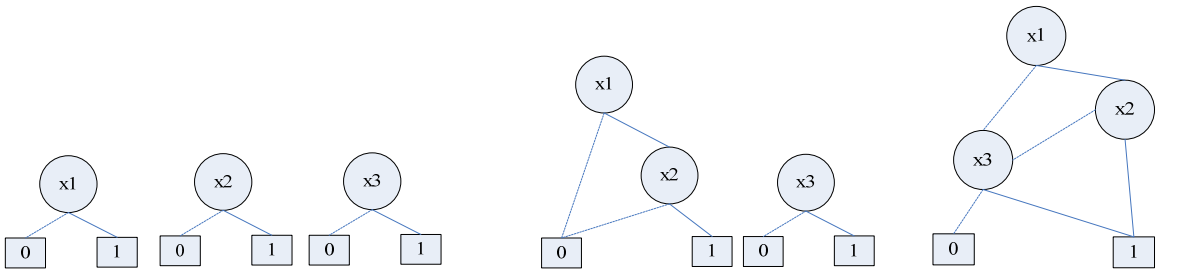
### 3.2.4 ROBDD 的建立

一般来说，实现电路中每个节点所代表的逻辑函数的 ROBDD，可以按照以下两步进行：

(1)对原始输入变量建立 ROBDD。

(2)从原始输入到原始输入，利用以上给出的算法逐级计算内部节点逻辑函数的 ROBDD 表示，直至到达原始输出。

下图 3-6 给出了一个简单的逻辑函数 $f_3=x_1x_2+x_3$ 的建立过程。




（a）对原始变量建立 ROBDD　　（b）得到乘积项 $x_1x_2$　　（c）最后完成和项之后

图 3-6 逻辑函数 $f_3$ 建立 ROBDD 树的过程

## 3.3 符号化矩量分析

在前面的章节中，分别介绍了矩的求解方式和符号化分析的重要算法 ROBDD（以下简称 BDD）算法。由于之前求解矩的过程都是基于数值方式的，因此很自然的就能联想到是否能用符号化的方式来求解矩，本小节将介绍利用符号化的方式来进行矩量分析。

### 3.3.1 算法的产生

假设现在有如下图 3-7 的简单 RC 树状电路。

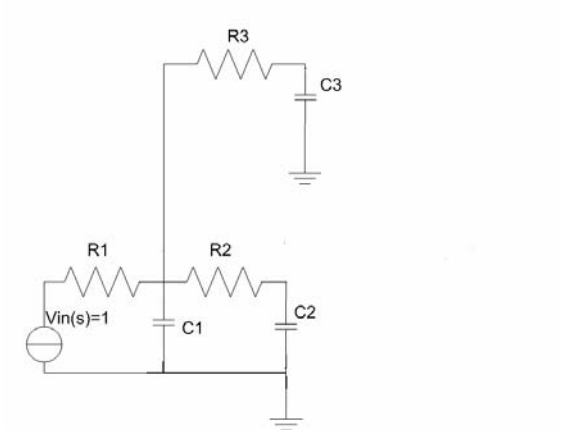



图 3-7 简单 RC 树状电路

通过上文介绍的方法，可以求得这个电路所有结点各阶的矩，现在列出 $m_0$、$m_1$、$m_2$ 三项。

| | $C_1$ | $C_2$ | $C_3$ |
|---|---|---|---|
| $m_0$ | 1 | 1 | 1 |
| $m_1$ | $(C_1+C_2+C_3)R_1$ | $(C_1+C_2+C_3)R_1+C_2R_2$ | $(C_1+C_2+C_3)R_1+C_3R_3$ |
| $m_2$ | $(C_1+C_2+C_3)^2R_1^2$ $+C_2^2R_1R_2+C_3^2R_1R_3$ | $(C_1+C_2+C_3)^2R_1^2+C_2^2R_1R_2+C_3^2R_1R_3$ $+(C_1+C_2+C_3)C_3R_1R_3+C_3^2R_3^2$ | $(C_1+C_2+C_3)^2R_1^2+C_2^2R_1R_2+C_3^2R_1R_3$ $+(C_1+C_2+C_3)C_2R_1R_2+C_2^2R_2^2$ |

表 3-2 简单 RC 树状电路各点的前 3 个矩的值

从表中，我们发现，不论矩的值多么复杂，在 RC 树状电路中，各阶的矩一直都是 R 和 C 这两个系列变量的积和形式。

考虑到上文介绍的 BDD 算法中，所代表的逻辑表达式也都是积和形式，因此提出利用 BDD 的结构来构造电路的拓扑结构。

### 3.3.2 算法的介绍

对于任意的 RC 树状电路，要求得其所有结点各阶的矩，由 RICE 算法可知，首先要进行一次反向深度优先遍历，求得各个结点的电压值，然后再进行一次正向的深度优先遍历来求得各个结点的电流值。

算法的主要步骤如下：

（1） 首先，通过一个反向遍历，对所有的电容 C 进行 ROBDD 的建立，一共建立$i$个结点，每个结点的子结点分别为 0 和 1，并且建立响应的部分和 ROBDD 树。

（2） 其次，通过一个正向遍历，对所有的电阻 R 也进行 ROBDD 的建立，它的正边为电容 C 的部分和记为$\sum C_i$，负边为前面连接的电阻，或者是 0（此时应为最前面的电阻 R，即$R_1$）。

经过这两个步骤，就可以建立一个完整的 RC 树状 BDD 机构图。

（3）对于结点$i$的第$j$阶矩，只要从这个 BDD 图上的第$i$个结点开始遍历，当遍历次数小于$j$时，所有的电容$C_i$的结点的正边都为该 BDD 结构图中对应的$R_i$，当遍历次数等于$j$时，所有电容$C_i$的结点的正边都为地点 1。

以上步骤中，关键所在就是 C 的部分和$\sum C_i$的确定。现假设对于一个 RC 树状电路，电容和电阻的序号（即下标）在任一支路按照升序排列。那么一般来说，对于固定电容$C_i$，它的部分和就是这个电容所在支路中，所有序号大于$i$的电容之和。比如下图 3-8 所示的一个树状电路，它的所有电容相对应的部分和如下表 3-3 所示:

| 电容 | 对应部分和 |
|---|---|
| $C_1$ | $C_1+C_2+C_3+C_4+C_5$ |
| $C_2$ | $C_2+C_3$ |
| $C_3$ | $C_3$ |
| $C_4$ | $C_4+C_5$ |
| $C_5$ | $C_5$ |

表 3-3 图 3-8 所示电路图各电容对应部分和

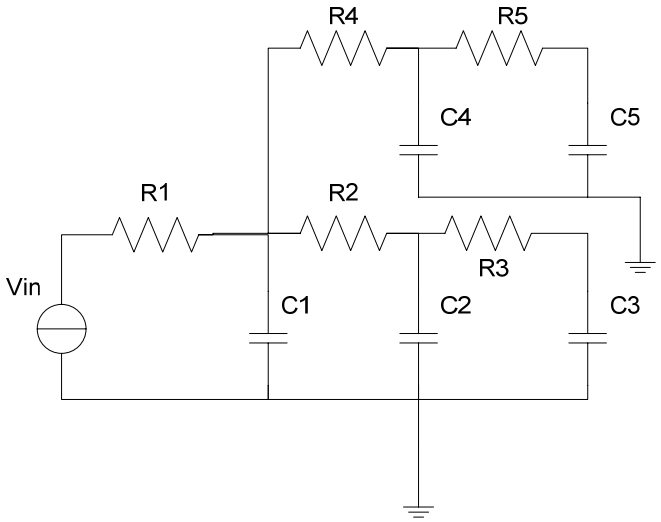



图 3-8 RC 树状电路图

对于如上的电路图，具体建立过程如下，首先是反向遍历，建立关于电容 C 的 BDD 结构。如下图 3-9 所示。

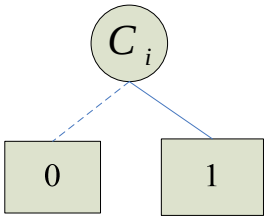
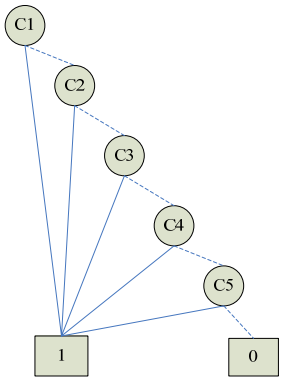
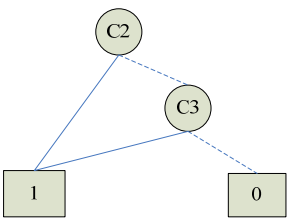



图 3-9 电容$C_i$的 ROBDD 机构图

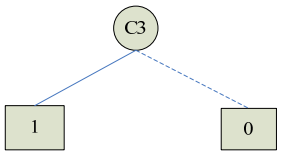
其中，$C_i$中的 i 分别为 1，2，3，4，5。而对于这 5 个电容 C，它们所对应的电容部分和的 BDD 结构如下图 3-10 所示。



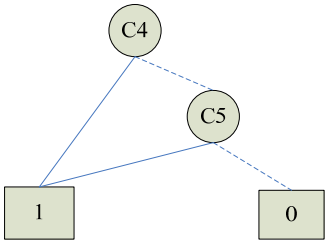

(a) $C_1$对应的部分和结构



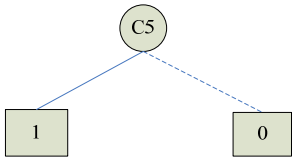

(b) $C_2$对应的部分和结构




(c) $C_3$对应的部分和结构




(d) $C_4$ 对应的部分和结构

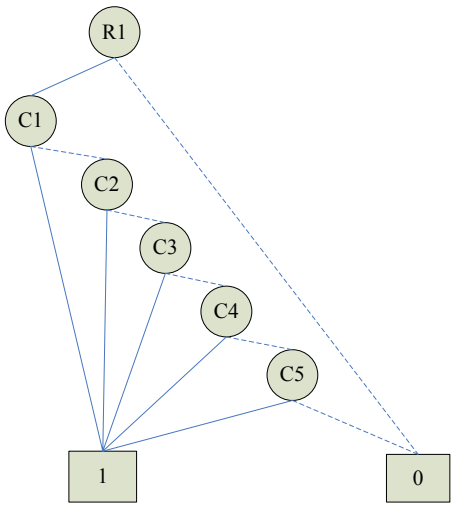



(e) $C_5$ 对应的部分和结构

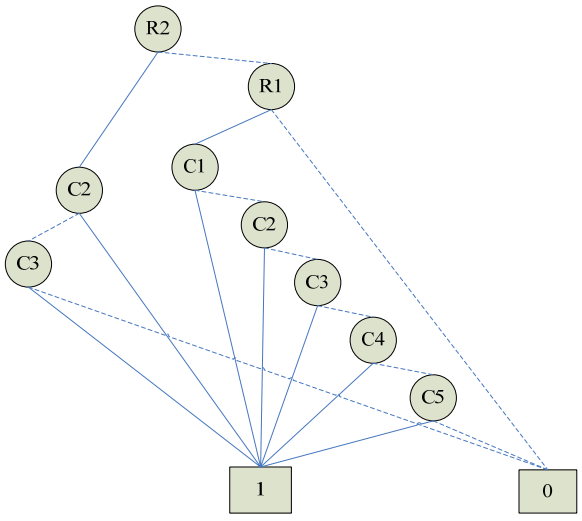
图 3-10　各电容 $C_i$ 所对应的部分和结构示意图

得到所有电容 $C_i$ 以及它对应的部分和的 BDD 结构图之后，第一步的反向遍历就结束了。接下来就是正向遍历确定各个电阻 $R_i$ 的对应的 BDD 结构图。整个正向遍历建立 BDD 结构图的过程如下图 3-11 所示。

建立的第一个结点是 $R_1$，它对应的电容部分和是 $C_1+C_2+C_3+C_4+C_5$，因此结构如图 3-11（a）所示。第二个结点是 $R_2$，它所对应电容的部分和为 $C_2+C_3$，根据前面建立步骤中介绍的，该结点的正边应当为它所对应的部分和，而它的负边应当为它前面的电阻，此时即为 $R_1$，那么此时建立 $R_2$ 之后的 BDD 结构如图 3-11(b)所示。




(a) $R_1$ 对应的 BDD 结构图




(b) $R_2$ 对应的 BDD 结构图

图 3-11　$R_1$ 与 $R_2$ 对应的 BDD 结构图

同理，可以建立 $R_3$ 的 BDD 结构图如下图 3-12 所示，图中，根据 BDD 的运算规则，由于 $R_3$ 对应的部分和为 $C_3$，为冗余项，因此 $R_3$ 正边直接连到了已有的结点上。

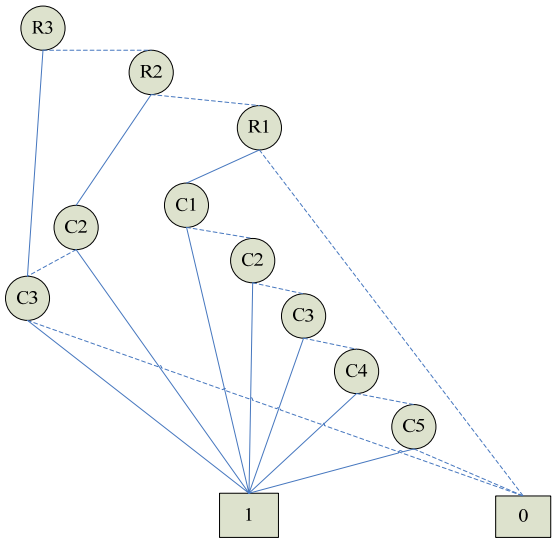



图 3-12 $R_3$ 的 BDD 结构图

根据电路图所示，$R_4$ 的前项电阻为 $R_1$，因此它的负边直接连接到结点 $R_1$，根据相同的原理，可以得到最终整个 RC 树状电路的 BDD 结构图，如下图 3-13 所示。

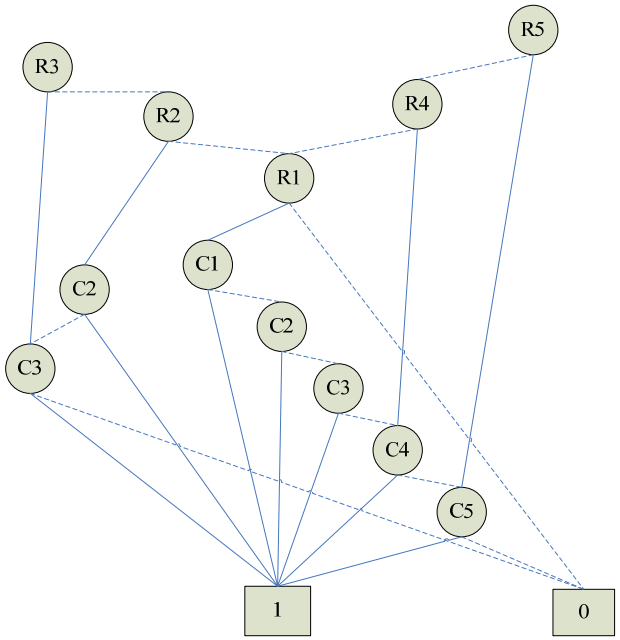



图 3-13 RC 树状电路 BDD 结构图

有了该电路的 BDD 结构图，按照之前介绍的步骤 3 就可以轻松的得到任意结点的任一阶矩的值。

仍旧是该电路图，假设需要知道电容$C_3$的一阶矩，可知只需要遍历 1 次，从结点$R_3$开始遍历，可以得到$C_3$的一阶矩的表达式为

$$R_3C_3 + R_2(C_2 + C_3) + R_1(C_1 + C_2 + C_3 + C_4 + C_5)$$

正好等于该结点的 Elmore 延时。

如果需要知道$C_3$电容的二阶矩，那么需要对 BDD 结构图遍历 2 次，从结点$R_3$开始遍历，当遍历次数等于 1 时，可以得到与刚才类似的表达式

$$R_3C_3' + R_2(C_2' + C_3') + R_1(C_1' + C_2' + C_3' + C_4' + C_5')$$

但由于遍历次数小于 2，此时所有的电容$C_i$的结点的正边都是都为该 BDD 结构图中对应的$R_i$，其实际含义就是指向了上一阶矩的值。因此，此时上式中的$C_i'$分别为：

$$C_1' = C_1R_1(C_1 + C_2 + C_3 + C_4 + C_5)$$

$$C_2' = C_2[R_2(C_2 + C_3) + R_1(C_1 + C_2 + C_3 + C_4 + C_5)]$$

$$C_3' = C_3[R_3C_3 + R_2(C_2 + C_3) + R_1(C_1 + C_2 + C_3 + C_4 + C_5)]$$

$$C_4' = C_4[R_4(C_4 + C_5) + R_1(C_1 + C_2 + C_3 + C_4 + C_5)]$$

$$C_5' = C_5[R_5C_5 + R_4(C_4 + C_5) + R_1(C_1 + C_2 + C_3 + C_4 + C_5)]$$

对于电容$C_3$的$i$阶矩，需要按照上面举例的步骤遍历$i$次电路的 BDD 结构图就可以计算得到。

在实际求解过程中，当代入数值运算时，并不需要每次都完全遍历整个过程，只需要将前一次计算得到的矩的值存起来，然后这次计算的时候直接调用即可。

### 3.3.3 算法的优化

在计算过程中，发现 BDD 结构图中的一些结点仍旧是冗余信息，比如图 3-14 中左边的 2 个结点。

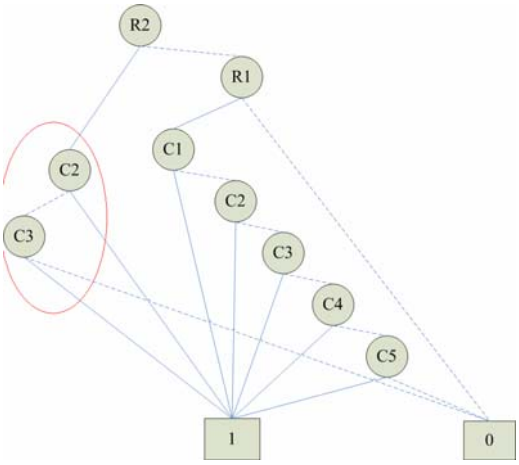



图 3-14 BDD 结构图及其冗余结点

在上面这个电路中，共有 5 个电容$C$，却出现了 2 个冗余的结点$C$。假设，现在共有$n$个电容$C$，分为 2 条支路，电路结构如图 3-15 所示，那么所产生的冗余结点数就是$n-2$个，BDD 结构如图 3-16 所示。

当支路再多点的情况下，冗余项的产生极为可怕，最坏的情况为，每多一条支路，冗余项增加个，此时冗余项个数与支路数的关系为：

$$num=\sum_{i=1}^{k}(n-2i)=kn-(k^2+k) \tag{3-4}$$

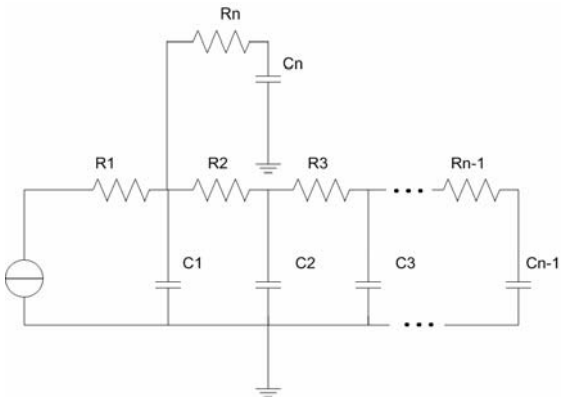



图 3-15 特殊的 RC 树状电路图

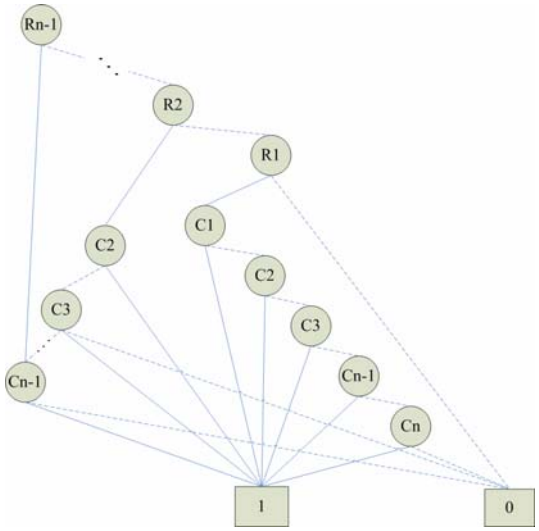



图 3-16 图 3-15 所示 RC 电路对应 BDD 结构图

对于一个含有 $n$ 个电容的 RC 树电路( $n$ 为偶数),最多有 $\frac{n}{2}+1$ 条支路,那么当 $k=\frac{n}{2}+1$ 时，总的冗余结点数为 $num=kn-(k^2+k)=\frac{1}{4}n^2-\frac{n}{2}-2$ (3-5)

为了去掉这些 $O(n^2)$ 冗余结点，就要对原来的 BDD 结构修改。新的电路结构图不再是单纯的 BDD 结构，而是一个 BDD 结构和树状结构的结合体。在上文提到的建立 BDD 图结构的第一步,原来是通过反向遍历建立所有电容 C 的 BDD 结构,如果此时仍旧采用 BDD 结构，那么不论如何优化算法，比如限定变量的输入顺序等等，或多或少都会产生不可避免的冗余结点。现在修改后，通过将这些电容建成一个树状结构，就可以完全避免冗余结点的产生。对于图 3-8 的电路图，树状的电容结构图如下图 3-17 所示。

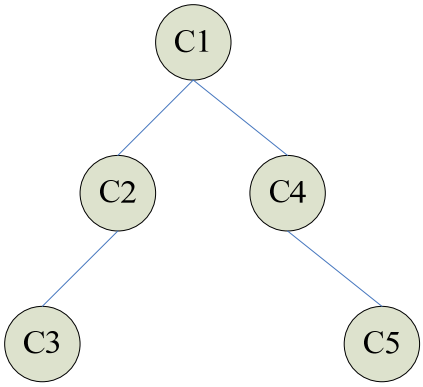



图 3-17 图 3-8 电路图中电容 C 对应的树状结构图

经过修改后的结构中,连接 2 个电容 C 结点间的边仅代表加法操作。那么对应图 3-8

电路的完整结构图如下图 3-18 所示：

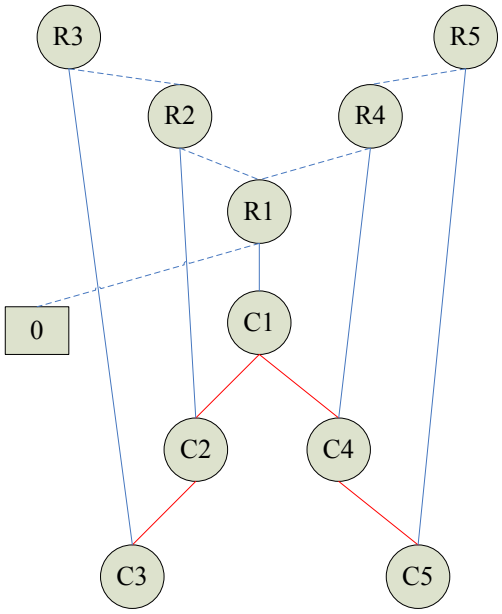



图 3-18 图 3-8 电路图对应结构图

优化后的算法，除了在第一步时，将原先的建立电容 C 的 BDD 结构改成了建立电容 C 的树状结构，后面 2 步均保持不变。求解矩的过程仍为原来的方法。

## 3.4 本章小结

本章先介绍了符号化理论中的重要算法，二叉判定图算法（BDD 算法）。通过分析介绍 BDD 算法的特性，发现 BDD 的结构正好符合典型的 RC 树状电路的结构，而其运算表示为乘加的特性正好符合电路矩的表达式的特性。因此随后提出了新的利用符号化分析电路矩的算法。通过例子具体介绍了这个符号化分析算法。再根据数据结构的优化，提出了最后的由 BDD 结构和树状结构相结合的符号化算法。

在下一章中，会就这种符号化算法在时序估计方面的效率进行分析。在第五章，将通过符号化算法来分析电路中的元素对于电路矩的影响。

# 第四章 延时应用

在 IC 设计过程中，不论是布局布线，还是插入缓存，都需要通过考虑 RC 延时来优化性能。至今为止，Elmore 延时仍十分被广泛使用就是因为它具有一个简单闭合的表达式。但随着电路设计工艺的特征尺寸越来越小，Elmore 延时已经逐渐不能对电路提供较高的精度，而这也就引起了众多学者对延时近似的研究。

## 4.1 时序估计

### 4.1.1 DM1 和 DM2 近似

假设 RC 树电路中，有结点集合$\{v_0,v_1,v_2,\cdots,v_n\}$，其中是$v_0$输入源，对于任意结点$v_i$（$0<i\le n$）有电容接地。另$p_i$为该结点的前向结点，或者说是父结点。而$R_i$为$p_i$和$v_i$之间的电阻。再令$R_{ki}$为路径$v_0$至$v_i$与路径$v_0$至$v_k$之间相重复的部分电阻和。那么，可以将结点$v_i$的 Elmore 延时，即路径$v_0$至$v_i$的 Elmore 延时写成如下表达式：

$$ED_i=\sum_{k=1}^{n}R_{ki}C_i \tag{4-1}$$

从公式中也可以看出，通过两次遍历就可以完全求出任意结点的 Elmore 延时。Elmore 延时也可以写成如下的递归形式：

$$ED_i=ED_{pi}+R_i(C_i+C_{di}) \tag{4-2}$$

其中，$C_{di}=\sum_{k\in s(i)}C_k$ (4-3)

$s(i)$是沿着结点$v_i$所在路径往后的所有结点的集合。

令$m_j^i$为结点$v_i$的第$j$阶矩。假设输入源为理想单位输入源，那么有$m_0^i=1$。这样通

过递归调用可以写出 $m_j^i$ 的表达式：

$$m_j^i = -\sum_{k=1}^{n} R_{ki} C_k m_{j-1}^k \tag{4-4}$$

通过矩匹配的方法，可以对一个电路的传输函数近似降阶成 $q$ 阶，有：

$$\hat{H}(s) = \frac{k_1}{s-p_1} + \frac{k_2}{s-p_2} + \cdots + \frac{k_q}{s-p_q} \tag{4-5}$$

其中 $k_1, k_2, \cdots, k_q$ 是极点 $p_1, p_2, \cdots, p_q$ 相对应的留数。对于一个 RC 树电路，在 s 平面上，所有的极点都处于负实数轴上。令 $V(s)$ 为单位阶跃输入的电压响应的 Laplace 变换。那么 $V(s) = \hat{H}(s) \bullet \frac{1}{s}$ (4-6)

它在时域上的响应 $v(t)$ 就可以写成：

$$v(t) = 1 + \frac{k_1}{p_1} e^{p_1 t} + \frac{k_2}{p_2} e^{p_2 t} + \cdots + \frac{k_q}{p_q} e^{p_q t} \tag{4-7}$$

进行尺度变换，令 $k_i' = \frac{k_i}{p_i}$，下面讨论中所有提到的 $k_i$ 就是这里的 $k_i'$。那么上式(4-7)可以写成如下简单形式：

$$v(t) = 1 + k_1 e^{p_1 t} + k_2 e^{p_2 t} + \cdots + k_q e^{p_q t} \tag{4-8}$$

利用上文提到的 AWE（渐进波形求值）算法，可以用电路各结点的前 $2q$ 阶矩求得 $q$ 阶的极点和留数。对于这样一个 $q$ 阶近似的模型降阶问题，所有极点满足 $0 \geq p_1 \geq p_2 \geq \cdots \geq p_q$，如果存在极点 $p_1 \gg p_2$ 这种情况，那么可以称 $p_1$ 为主极点。

如果是 $p_1$ 主极点，对于式(4-8)可以忽略后面 $q-1$ 项，近似为：

$$v(t) = 1 + k_1 e^{p_1 t} \tag{4-9}$$

只要令 $v(t)$ 等于 0.5，就可以解得该结点 50%的延时：

$$t_D = -\frac{1}{p_1}\ln(2k_1) \tag{4-10}$$

此时 $q=1$，再令 $k_1=1$ 有 $p_1=\frac{1}{m_1}$，那么此时式(4-10)就可以写成：

$$t_D = -m_1\ln(2) \tag{4-11}$$

明显的，上式(4-11)是 Elmore 延时的一个尺度变换。

当考虑式(4-8)含有 2 个极点，即 $p_1$ 和 $p_2$ 时，假定 $v(t=0)=0$，它在时域的传输函数就是

$$v(t) = 1 + k_1 e^{p_1 t} + k_2 e^{p_2 t} \tag{4-12}$$

通过用前 3 个矩 $m_0, m_1, m_2$ 可以求得这两个极点：

$$p_{1,2} = \frac{2}{m_1 \mp \sqrt{4m_2 - 3m_1^2}} \tag{4-13}$$

而留数为：

$$k_1 = -k_2 = -\frac{1}{\sqrt{4m_2 - 3m_1^2}} \tag{4-14}$$

再将式(4-13)和式(4-14)代入式(4-12)可以求得此时的延时，就是 DM1：

$$DM1 = \frac{1}{2}(-m_1 + \sqrt{4m_2 - 3m_1^2})\ln(1 - \frac{m_1}{\sqrt{4m_2 - 3m_1^2}}) \tag{4-15}$$

为了简化 DM1 公式，将这两个主极点 $p_1$、$p_2$ 近似成一个主极点 $p_q$：

$$\frac{1}{p_d} = -\sqrt{\frac{1}{p_1^2} + \frac{1}{p_2^2}} = -\sqrt{2m_2 - m_1^2} \tag{4-16}$$

令 $k=1$，那么此时的延时表示就是 DM2：

$$DM2 = \sqrt{2m_2 - m_1^2}\ln(2) \tag{4-17}$$

根据相同的方法，可以求得 DM3，DM4 的表达近似，这里就不再一一描述其推导

过程。

### 4.1.2 D2M 近似

从上面 DM1 和 DM2 公式可以看出，前 3 阶的矩可以提供足够的信息来对延时进行较为精确的近似。但是，不论是那种方法，都需要通过大量的推导计算工作来求得延时值。

这里，要介绍由 Charles.J.Alpert 提出的一个经验公式：

$$D2M = -\frac{m_1}{\sqrt{m_2}}(-m_1)\ln(2) = \frac{m_1^2}{\sqrt{m_2}}\ln(2) \tag{4-18}$$

Gupta 证明过对于一个 RC 树电路，它的传输函数的二阶中心矩一定是非负的，即

$$\mu_2 = 2m_2 - m_1^2 \geq 0 \tag{4-19}$$

那么，对于式(4-18)中的 D2M，必定是小于 Elmore 延时的，因为：

$$D2M = \frac{m_1^2}{\sqrt{m_2}}\ln(2) = m_1\ln(2)\sqrt{\frac{m_1^2}{m_2}} \leq m_1\ln(2)\sqrt{2} = 0.9802m_1 \tag{4-20}$$

但需要注意的是，式(4-20)所表达的意义并不只是 D2M 比 Elmore 延时小 2%，而是它的上限相对的比 Elmore 延时的上限小了 2%，因此它比 Elmore 延时更为精确。

## 4.2 试验数据

### 4.2.1 测试延时近似的精度

#### 4.2.1.1 当仅存在一条支路的情况

电路图如图 4-1 所示。

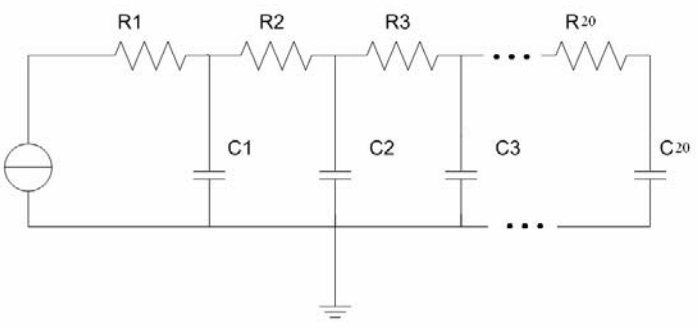


图 4-1 包含 20 个电阻和电容的 RC 树状电路图

其中，各元件值分别为：

$R_i = 80\Omega$， $i = 1,2,\ldots,20$

$C_i = 1pF$， $i = 1,2,\ldots,20$

经过测试得到的延时比较：

| | SPICE | Elmore | D2M | DM2 |
|---|---|---|---|---|
| $C_1$ | 0.11 | 1.60 | 0.41 | 4.05 |
| $C_2$ | 0.36 | 3.12 | 1.12 | 5.52 |
| $C_3$ | 0.80 | 4.56 | 1.95 | 6.52 |
| $C_4$ | 1.42 | 5.92 | 2.86 | 7.25 |
| $C_5$ | 2.22 | 7.20 | 3.80 | 7.81 |
| $C_6$ | 3.18 | 8.40 | 4.75 | 8.24 |
| $C_7$ | 4.30 | 9.52 | 5.69 | 8.58 |
| $C_8$ | 5.20 | 10.56 | 6.60 | 8.84 |
| $C_9$ | 6.71 | 11.52 | 7.48 | 9.03 |
| $C_{10}$ | 7.81 | 12.40 | 8.31 | 9.18 |
| $C_{11}$ | 8.78 | 13.20 | 9.08 | 9.30 |
| $C_{12}$ | 9.63 | 13.92 | 9.79 | 9.38 |
| $C_{13}$ | 10.36 | 14.56 | 10.43 | 9.43 |
| $C_{14}$ | 10.98 | 15.12 | 11.00 | 9.47 |
| $C_{15}$ | 11.49 | 15.60 | 11.49 | 9.49 |
| $C_{16}$ | 11.92 | 16.00 | 11.91 | 9.51 |
| $C_{17}$ | 12.25 | 16.32 | 12.24 | 9.52 |
| $C_{18}$ | 12.51 | 16.56 | 12.50 | 9.52 |
| $C_{19}$ | 12.65 | 16.72 | 12.67 | 9.52 |
| $C_{20}$ | 12.73 | 16.80 | 12.75 | 9.52 |

表 4-1 图 4-1RC 电路的各种延时近似和 SPICE 及 Elmore 延时的比较（单位$10^{-9}s$）

从上表中，可以看出来，经验公式 D2M 在远端的近似几乎与 SPICE 仿真器一致，而在近端处稍微有点偏大。而 Elmore 延时不论在近端和远端都比 SPICE 仿真要大很多，可见 D2M 在近似延时方面要比单纯的使用 Elmore 延时要精确。而 DM2 公式，由于是近似了一个主极点，因此在电路的中间部分，较为精确，和 SPICE 仿真结果差别不大，但是在远端和近端都有一定的差距，近端偏大，远端偏小。

#### 4.2.1.2 多条支路的情况

电路图如图 4-2 所示。

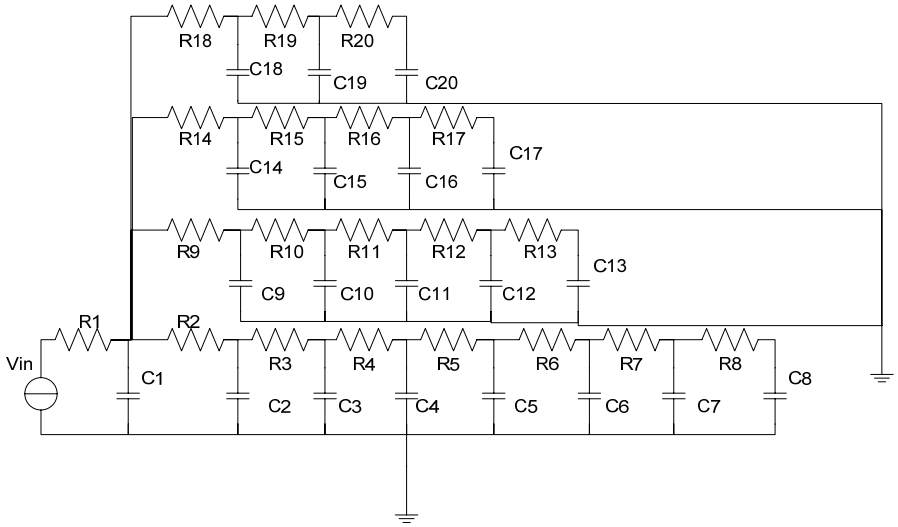



图 4-2 有多条支路的 RC 树状电路图

其中，各元件值分别为：

$R_1 = 100\Omega$， $R_2 = R_3 = R_4 = R_5 = R_6 = R_7 = R_8 = 80\Omega$

$R_9 = R_{10} = R_{11} = R_{12} = R_{13} = 70\Omega$， $R_{14} = R_{15} = R_{16} = R_{17} = 90\Omega$

$R_{18} = R_{19} = R_{20} = 60\Omega$

$C_1 = C_2 = C_3 = C_4 = C_5 = C_6 = C_7 = C_8 = 0.8pF$

$C_9 = C_{10} = C_{11} = C_{12} = C_{13} = 1pF$， $C_{14} = C_{15} = C_{16} = C_{17} = 0.9pF$

$C_{18} = C_{19} = C_{20} = 1.2pF$

经过测试得到的延时比较。

| | SPICE | Elmore | D2M | DM2 |
|---|---|---|---|---|
| $C_1$ | 0.89 | 1.86 | 1.08 | 1.75 |
| $C_2$ | 1.39 | 2.31 | 1.47 | 1.87 |
| $C_3$ | 1.83 | 2.69 | 1.83 | 1.95 |
| $C_4$ | 2.19 | 3.01 | 2.14 | 1.99 |
| $C_5$ | 2.49 | 3.27 | 2.40 | 2.01 |
| $C_6$ | 2.67 | 3.46 | 2.59 | 2.02 |
| $C_7$ | 2.81 | 3.59 | 2.73 | 2.02 |
| $C_8$ | 2.87 | 3.65 | 2.80 | 2.02 |
| $C_9$ | 1.33 | 2.21 | 1.40 | 1.80 |
| $C_{10}$ | 1.65 | 2.49 | 1.67 | 1.83 |
| $C_{11}$ | 1.89 | 2.70 | 1.89 | 1.84 |
| $C_{12}$ | 2.03 | 2.84 | 2.03 | 1.85 |
| $C_{13}$ | 2.11 | 2.91 | 2.10 | 1.85 |
| $C_{14}$ | 1.33 | 2.18 | 1.38 | 1.78 |
| $C_{15}$ | 1.58 | 2.43 | 1.62 | 1.80 |
| $C_{16}$ | 1.76 | 2.59 | 1.79 | 1.81 |
| $C_{17}$ | 1.84 | 2.67 | 1.87 | 1.81 |
| $C_{18}$ | 1.16 | 2.08 | 1.29 | 1.76 |
| $C_{19}$ | 1.31 | 2.22 | 1.43 | 1.76 |
| $C_{20}$ | 1.38 | 2.29 | 1.50 | 1.76 |

表 4-2 图 4-2RC 电路图的各种延时近似和 SPICE 及 Elmore 延时的比较（单位$10^{-9}s$）

该测试总电容数和上个测试一样，都是 20，但是由于支路个数增加，每个支路所含的电容数减少，因此在每个支路的远端，都没有和上个例子一样精确。但总体来说，对于经验公式 D2M，在任一支路的远端，它的精确度都是最好的；Elmore 延时由于只是一阶矩的值，因此整体仍旧偏大；而 DM2 在每个支路的中段近似的较好，近端偏大，远端偏小。

### 4.2.2 测试延时近似的速度

这个测试主要用来验证程序能运算和建立多大的电路。

现将一个电阻 R 和一个电容 C 构成的结构称为一个梯形结构,对于规则的 RC 树状电路，有多少个这样的梯形结构就能决定该电路有多大。运行硬件环境是：

芯片 Intel Pentium 1.7GHz，内存 1G

下表 4-3 为运行电路所含梯形结构的个数及其对应运行的时间，该运行时间包括读取电路网表结构，建立对应电路结构图以及分别计算一阶和二阶矩。

| 电路图（RC） | 一阶矩 | 二阶矩 |
|---|---|---|
| 4 个 梯形结构 | 0.015 | 0.001 |
| 30 个梯形结构 | 0.016 | 0.001 |
| 100 个梯形结构 | 0.046 | 0.022 |
| 500 个梯形结构 | 0.703 | 0.372 |
| 2000 个梯形结构 | 17.656 | 7.422 |
| 4000 个梯形结构 | 99.969 | 27.468 |
| 10000 个梯形结构 | 1449.551 | 203.517 |

表 4-3 电路图大小及其对应运算时间（单位：秒）

上表中，计算二阶矩是建立在一阶矩的基础上，其所用时间较少是因为少了读取电路网表并建立对应结构图。从上表中，不难看出，这种算法可以处理较大的 RC 树状电路，速度较快。

在分析电路延时时，利用符号化分析能快速的计算矩，有效率的近似电路延时。下表 4-4 列出了本算法和 SPICE 的速度比较。近似延时采用了经验公式 D2M。

| 电路图（RC） | 符号化分析 | HSPICE |
|---|---|---|
| 4 个 梯形结构 | 0.016s | 0.017s |
| 30 个梯形结构 | 0.017s | 2.33s |
| 100 个梯形结构 | 0.068s | 15.68s |
| 500 个梯形结构 | 1.075s | 72.35s |
| 2000 个梯形结构 | 25.078s | 532.67s |
| 4000 个梯形结构 | 127.437s | 1435.24s |
| 10000 个梯形结构 | 1653.068s | ～ |

表 4-4 符号化分析近似延时和 HSPICE 分析延时速度比较

当电路不大时，差距并不明显，当电路达到上千个梯形结构时，HSPICE 运行速度明显很慢，而符号化分析的效率就较为客观，在近似电路延时的时候，只需要建立结构并且计算得到前 2 阶矩即可。

## 4.3 本章小结

本章介绍了几种较为精确的近似延时的方法，并对这些方法进行了测试和分析，利用符号化分析方法，可以快速的得到各阶的矩和电路延时。通过试验数据，选择不同的近似公式，可以在不同的地方得到相对较为精确的结果。比如 D2M，在远端十分精确，几乎和 SPICE 仿真一样，而在近端相差较大。而 DM2，在近端和远端都不是十分精确，但在电路的中间部分却相对精确。因此，考虑需要分析的结点的位置，选择一个适当的近似公式。

在测试精度的同时，还利用符号化算法和 SPICE 仿真的测试速度进行了比较。很明显的，由于算法采用了较为简单的数据结构，因此可以迅速的计算上万阶的电路的矩，只需要建立一个电路结构即可，而 SPICE 需要通过瞬态分析来计算延时。因此，在速度方面，符号化分析法有很明显的优势。

除了以上提到的速度和精度，符号化分析法可以直接观察到各个结点任一阶矩的表达式，而这可以使我们观察到电路中各元件对于某一固定结点的矩的影响，在下一章中，会就这一问题展开详细的讨论。

# 第五章 敏感性分析

利用符号化分析可以得到 RC 树状电路的结构图，并且可以获得任一结点的任意阶矩的表达式。如果代入数值，也能轻易的获得各个结点的矩的数值，本章将讨论这些电路矩的敏感性。

现假定 $D(i)$ 为沿着结点 $i$ 所在支路，自结点 $i$ 开始，所有序号大于 $i$ 的结点。 $P(n)$ 为从结点 $n$ 至输入端的路径。 $m_i^j$ 为第 $i$ 个结点的第 $j$ 阶矩。

那么对于任意结点 $i$ 的第 $j$ 阶矩，我们可以将它表示成

$$m_i^j = \sum_{l \in P(i)} (R_l \sum_{k \in D(i)} C_k m_k^{j-1}) \tag{5-1}$$

对于这个公式，如果不考虑公式的迭代性，观察发现其中主要有 3 个参数，即电阻 R，电容 C 和上一阶的矩 $m_i^{j-1}$，那分别对这 3 者求偏导就可以得出矩对这 3 个参数的敏感性。现在假设 $w_l$ 是这个互连线的宽度，那么有如下偏导：

$$\frac{\partial m_i^j}{\partial w_l} = \frac{\partial m_i^j}{\partial R_l}\frac{\partial R_l}{\partial w_l} + \frac{\partial m_i^j}{\partial C_l}\frac{\partial C_l}{\partial w_l} + \sum_k \frac{\partial m_i^j}{\partial m_k^{j-1}}\frac{\partial m_k^{j-1}}{\partial w_l} \tag{5-2}$$

从上式中，不难发现对于电路矩，它的敏感度是十分复杂的，因为对其产生影响的不仅仅是电路元件电阻和电容，还包括动态变化的电路各个结点矩对其的影响。因此，要十分精确的分析电路矩的敏感度是十分困难的。在这里，将电路的上一阶矩近似看成电路的元件，那么上面公式中的三部分分别对电路矩产生了影响，下面依次对其进行讨论。

## 5.1 对于电阻的敏感度

电阻 R 是构成 RC 树状电路的重要组成部分，在电路不同地方的电阻对于固定结点的第 $j$ 阶矩的影响是不同的。

对于电路中任意的电阻 R，有：

$$\frac{\partial m_i^j}{\partial R_k} = (-1)^j M_{j-1}^k, \quad i \in D(k) \tag{5-2}$$

其中，$M_k^{j-1} = \sum_{l \in D(i)} C_l (-1)^{j-1} m_l^{j-1}$

$$\frac{\partial m_i^j}{\partial R_k} = 0, \quad i \notin D(k) \tag{5-3}$$

上式(5-2)(5-3)说明，对于固定结点$i$，任一结点$k$对它的$j$阶矩的影响，是沿着结点$i$所在支路，自结点$i$开始，所有序号大于$i$的结点的电容与上一阶矩的乘积的和。由于上式中还包含迭代的上一阶矩，因此这并不是完全准确的敏感度，而是近似的敏感度。下文讨论的电容和前一阶矩对当前矩的影响都是如此。

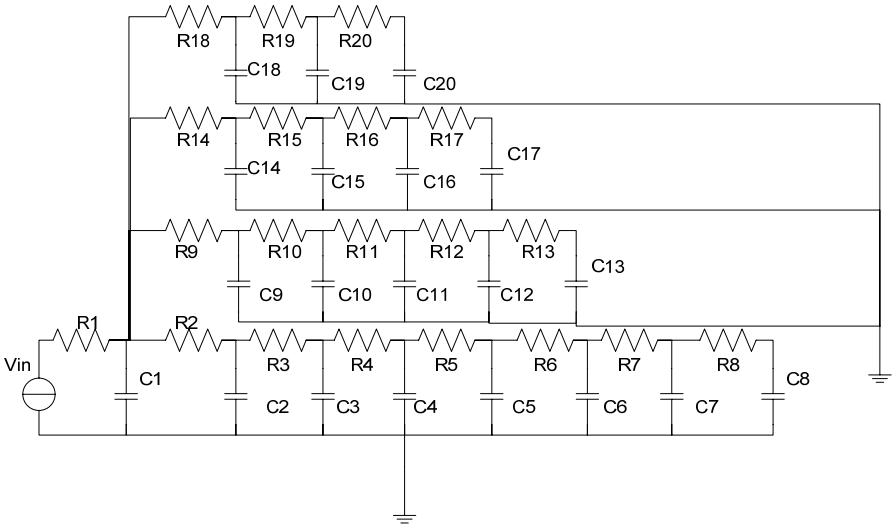



图 5-1 RC 电路图

对于上图中结点 8，连接电容$C_8$，其余各电阻值均为$100\Omega$，电容值均为$1pF$。通过改变电阻$R_8$的阻值观察三阶矩$m_8^3$与电阻的关系，下表 5-1 为三阶矩，二阶矩，一阶矩和阻值的关系。

| $R_8$ 的阻值（Ω） | 一阶矩 $m_8^1(10^{-9})$ | 二阶矩 $m_8^2(10^{-18})$ | 三阶矩 $m_8^3(10^{-27})$ |
|---|---|---|---|
| 10 | 4.71 | 18.07 | 65.7 |
| 20 | 4.72 | 18.13 | 65.9 |
| 50 | 4.75 | 18.29 | 66.7 |
| 75 | 4.78 | 18.43 | 67.2 |
| 100 | 4.80 | 18.57 | 67.87 |
| 150 | 4.85 | 18.85 | 69.11 |
| 200 | 4.90 | 19.14 | 70.39 |
| 300 | 5.00 | 19.73 | 73.05 |
| 500 | 5.20 | 20.97 | 78.79 |
| 750 | 5.45 | 22.63 | 86.82 |
| 1000 | 5.70 | 24.42 | 95.90 |
| 1500 | 6.20 | 28.37 | 117.56 |
| 2000 | 6.70 | 32.82 | 144.52 |
| 3000 | 7.70 | 43.22 | 217.34 |
| 5000 | 9.70 | 70.02 | 459.58 |
| 10000 | 14.70 | 172.02 | 1908.68 |

表 5-1 阻值和矩的关系

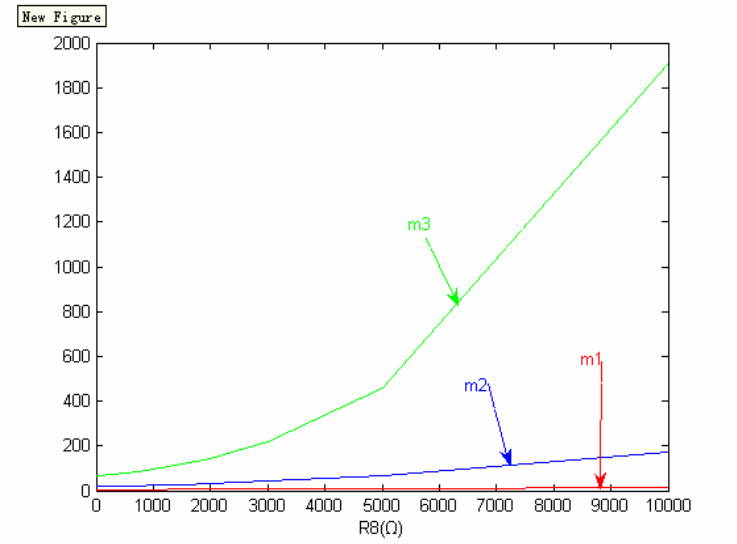



图 5-2 矩和阻值的关系图

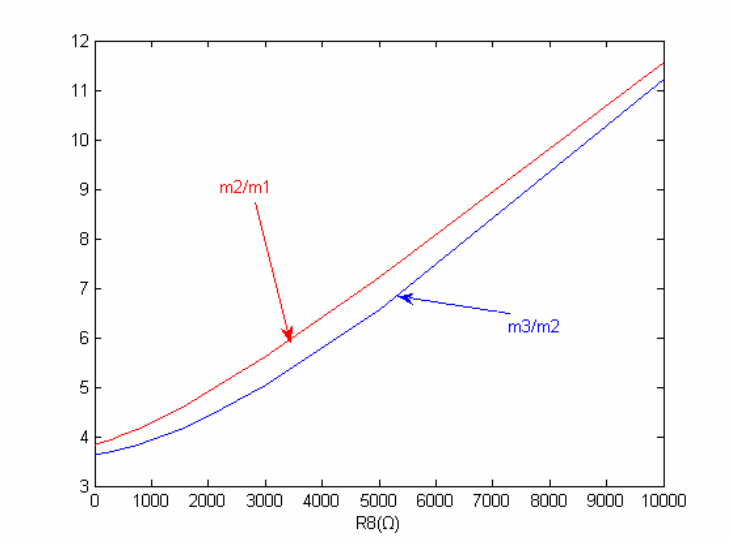



图 5-3 阻值和前后 2 阶矩的比例之间的关系

图 5-3 中，红色曲线代表的是结点 8 的二阶矩和一阶矩的比值与电阻 R8 的关系，蓝色曲线代表的是结点 8 的三阶矩和二阶矩的比值与电阻 R8 的关系。

仅仅从图 5-2 很难发现阻值和矩之间的关系，只能判断出随着阶数的增高，阻值的变化对矩的影响越来越大。图 5-2 中的最下面的曲线代表的是，一阶矩根据电阻 R8 变化的曲线。从 0 至 10000 间比较平稳，几乎不变，而当中的曲线，代表的是二阶矩，是效率较小的线性曲线，至于最上方的曲线，代表的是三阶矩，到了后期，几乎是按平方变化。这一点可以从图 5-3 中得到解释，随着阻值的增大，前后 2 阶矩之间的比值是线性增大的，那么对于 $m_3/m_1$ 显然就是平方变化了。

| $R_8$ 的阻值（Ω） | 一阶矩 $m_8^1(10^{-9})$ | 二阶矩 $m_8^2(10^{-18})$ | 三阶矩 $m_8^3(10^{-27})$ |
|---|---|---|---|
| 10 | 4.7100 | 18.0741 | 65.7088 |
| 10.1 | 4.7101 | 18.0746 | 65.7112 |
| 10.2 | 4.7102 | 18.0752 | 65.7135 |
| 10.3 | 4.7103 | 18.0757 | 65.7158 |
| 10.4 | 4.7104 | 18.0763 | 65.7182 |
| 10.5 | 4.7105 | 18.0768 | 65.7205 |
| 10.6 | 4.7106 | 18.0774 | 65.7229 |
| 10.7 | 4.7107 | 18.0779 | 65.7252 |
| 10.8 | 4.7108 | 18.0784 | 65.7256 |

| | | | |
|---|---|---|---|
| 10.9 | 4.7109 | 18.0790 | 65.7299 |
| 11 | 4.7110 | 18.0795 | 65.7323 |
| 11.1 | 4.7111 | 18.0801 | 65.7346 |
| 11.2 | 4.7112 | 18.0806 | 65.7370 |
| 11.3 | 4.7113 | 18.0811 | 65.7393 |
| 11.4 | 4.7114 | 18.0817 | 65.7416 |
| 11.5 | 4.7115 | 18.0822 | 65.7440 |

表 5-2 阻值和矩的关系

从上表 5-2 中，可以看出前 3 阶矩几乎是随着电阻 $R_8$ 的改变而线性增长的。斜率几乎是上一阶矩的值。

以上研究的是在同一支路上的电阻对矩的影响。下表 5-3 列出了不同支路的电阻对矩的影响。

| $R_{13}$ 的阻值（Ω） | 一阶矩 $m_8^1(10^{-9})$ | 二阶矩 $m_8^2(10^{-18})$ | 三阶矩 $m_8^3(10^{-27})$ |
|---|---|---|---|
| 50 | 4.80 | 18.57 | 67.84 |
| 100 | 4.80 | 18.57 | 67.87 |
| 250 | 4.80 | 18.59 | 67.96 |
| 500 | 4.80 | 18.61 | 68.12 |
| 1000 | 4.80 | 18.66 | 68.44 |
| 2000 | 4.80 | 18.76 | 69.07 |
| 5000 | 4.80 | 19.06 | 70.96 |
| 10000 | 4.80 | 19.56 | 74.11 |

表 5-3 不同支路电阻对矩的影响

从上表中，不难发现，由于 $R_{13}$ 和结点 8 并不处于同一支路，因此对于结点 8 的矩的影响几乎为 0。所有矩的值几乎不变。所以当两个结点处于不同的支路时，互相之间没有影响。

## 5.2 对于电容的敏感度

除了电阻 R，电容 C 在 RC 树状电路中是另外一个重要的组成部分。经过上面的讨论，已经知道电阻和前后矩的比值存在线性关系，下面就来讨论电容 C 和矩值的关系。

对于电路中任意的电容 C，有：

$$\frac{\partial m_i^j}{\partial C_k} = -\sum_{l \in P(i) \cap P(k)} R_l m_l^{j-1} \tag{5-4}$$

这个公式说明任意结点的矩的值和任一电容的关系是，从输入源点至这两者之间的路径的公共部分的电阻和其上一阶矩的乘积的和。

还是看上面图 5-1 的 RC 树电路，假设所有电阻为 100，电容除了 $C_2$ 之外都是 $1pF$ 。考虑电容 $C_2$ 对结点 8 的影响，由于结点 2 至源点的路径和结点 8 至源点路径的公共部分为 $R_1$ 和 $R_2$，所以应该考虑这两项。下表 5-4 为电容和其因素与矩的关系。

| $C_2$ (pF) | $R_1 m_1^1 + R_2 m_2^1$ $(10^{-7})$ | $m_8^2$ $(10^{-18})$ | $R_1 m_1^2 + R_2 m_2^2$ $(10^{-16})$ | $m_8^3$ $(10^{-27})$ |
|---|---|---|---|---|
| 0.5 | 4.55 | 17.90 | 14.85 | 64.42 |
| 0.75 | 4.63 | 18.23 | 15.21 | 66.12 |
| 1 | 4.70 | 18.57 | 15.58 | 67.87 |
| 1.5 | 4.85 | 19.27 | 16.34 | 71.49 |
| 2 | 5.00 | 19.98 | 17.13 | 75.30 |
| 5 | 5.90 | 24.69 | 22.50 | 102.36 |
| 10 | 7.40 | 34.14 | 33.85 | 166.17 |
| 20 | 10.40 | 59.04 | 65.55 | 385.59 |
| 50 | 19.40 | 181.74 | 232.65 | 2210.25 |
| 100 | 34.40 | 546.24 | 751.15 | 12019.35 |
| 200 | 64.40 | 1875.24 | 2688.15 | 78617.55 |
| 500 | 154.40 | 10662.24 | 15699.15 | 1086252.15 |

表 5-4 RC 电路中电容对矩的影响

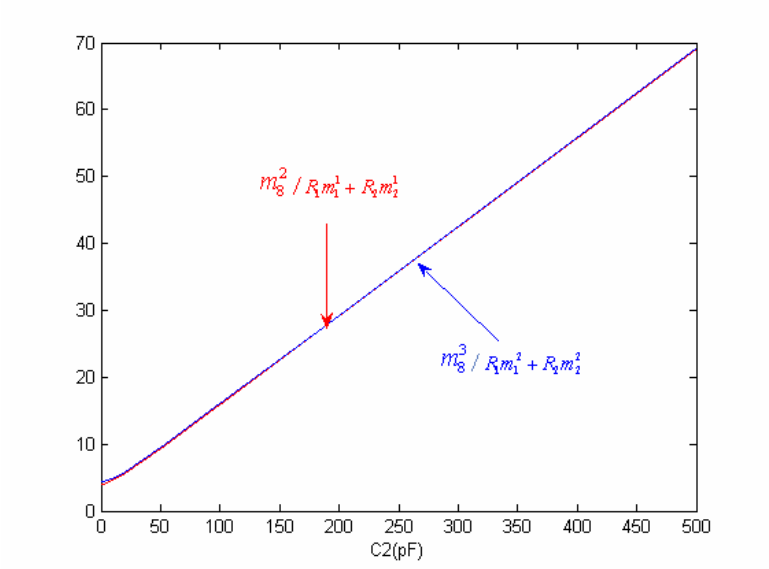



图 5-4 矩和电容之间的关系

由图 5-4 可以看出，电容$C_2$对结点 8 的矩的都是和他们至源点的公共部分与其上一阶的矩的乘积的和。而且对于固定的电容值$C_2$，这个比值是固定的。随着电容$C_2$的增大，这个比值线性增加。

但不要认为电容对于矩的影响是线性的。由于上图中牵涉到了上一阶的矩，因此这个迭代过程会让电容的影响放大数倍，下图是电容$C_2$对结点 8 的矩的影响。

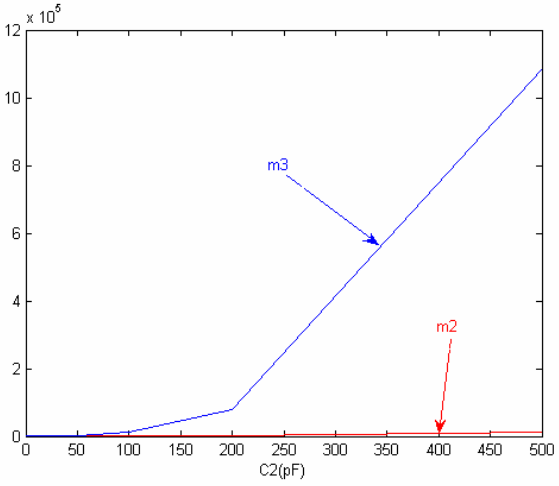



图 5-5 电容$C_2$与结点 8 的矩的直接关系图

图中，上方曲线代表的三阶矩对于电容 C 的变化十分敏感，对于更高阶的矩，敏感

度更加强。

## 5.3 对于前一阶矩的敏感度

从前面的分析可以看出，不论是电阻 R 还是电容 C，和某个结点的 $j$ 阶矩的关系中总会出现 $j-1$阶矩，下面就讨论下前一阶矩对这阶矩的影响。

对于电路中任意结点的第 $j-1$阶矩，有：

$$\frac{\partial m_i^j}{\partial m_k^{j-1}} = -\sum_{u \in P(i) \cap P(k)} R_u C_k \tag{5-5}$$

式(5-5)说明结点 $i$ 的 $j$ 阶矩和结点 k 的 $j-1$阶矩的比值，满足他们至源点的公共部分的 R 和结点 k 所连接电容 $C_k$ 的乘积的和。

继续看图 5-1 所示电路图，假设所有电阻为 100，电容都是$1pF$ 。考虑结点 3 的矩对结点 8 的矩的影响，由于结点 3 至源点的路径和结点 8 至源点路径的公共部分为 $R_1$、$R_2$ 和 $R_3$，所以应该考虑这三项。

通过改变不同支路上的电阻电容值，达到改变结点 3 的矩。保持 $R_1$、$R_2$ 和 $R_3$ 以及 $C_3$ 不变。

下表 5-4 列出了结点 3 和结点 8 之间的关系：

| $m_3^1\ (10^{-9})$ | $m_3^2\ (10^{-18})$ | $m_8^2\ (10^{-18})$ | $m_8^3\ (10^{-27})$ |
|---|---|---|---|
| 3.30 | 11.72 | 18.57 | 67.87 |
| 3.35 | 12.03 | 18.96 | 69.56 |
| 3.40 | 12.35 | 19.35 | 71.30 |
| 3.70 | 14.48 | 21.93 | 82.84 |
| 4.00 | 16.97 | 24.87 | 96.25 |
| 4.70 | 24.18 | 33.13 | 134.83 |

| | | | |
|---|---|---|---|
| 6.20 | 46.23 | 57.43 | 251.82 |
| 8.20 | 89.63 | 103.83 | 480.60 |
| 13.20 | 268.13 | 289.83 | 1416.55 |
| 23.20 | 925.13 | 961.83 | 4848.45 |

表 5-6 部分结点的矩与结点 8 的矩的关系

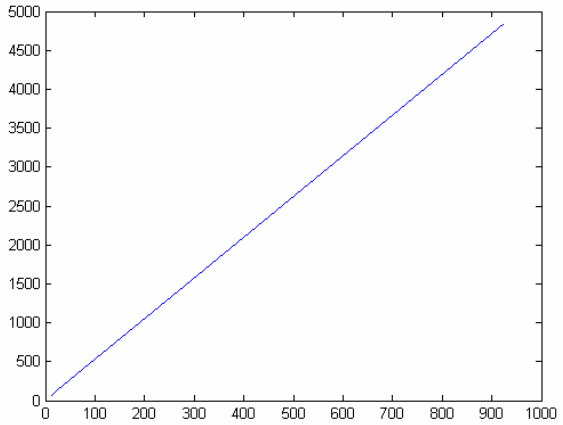



图 5-6 结点 8 的 3 阶矩与结点 3 的 2 阶矩之间关系图

从图中可以看出，结点 8 的 3 阶矩与结点 3 的 2 阶矩的关系成线性关系，这是因为前提为保证 $R_1$、$R_2$ 和 $R_3$ 以及 $C_3$ 不变的情况下。若通过更改前者中的某一项值来改变结点 3 的 2 阶矩的值，那么 $m_8^3$ 与 $m_3^2$ 之间就不再是简单的线性关系了。

## 5.4 敏感性测试的作用

随着超大规模集成电路工艺的迅速发展人们已能制造越来越复杂的芯片，这些芯片通过千百条互连线把器件和部件连接起来形成像微处理器门阵列或信息处理和高速计算这样复杂的芯片为了不断提高芯片速度和芯片的集成度，近年来采用了越来越多层次的互连线研究表明在高速高密度集成电路中限制其发展的主要因素不是器件的门延迟而是互连线的寄生元件引起的时间延迟互连线之间信号的串扰和电路功耗。芯片内部和芯片之间的互连线在决定一个数字系统的物理尺寸，功耗以及时钟频率方面起着越来越重要的作用。特别是由于亚微米、深亚微米工艺技术和芯片集成、圆片集成技术的出现，使得芯片上或圆片上高密度互连线的寄生参数，如电阻、电容、电感以及由此引起的信号传播延时和信号串扰已成为设计极高速超大规模集成电路的一个主要考虑因素。

在微电子电路中，特征尺寸按比例缩小在提高电路性能和减小芯片单位功能成本以提高产量方面颇见成效。电路速度的增长主要是由于晶体管门长度的减小使开关速度提高。当尺寸缩小到亚微米时，由较高 RC 乘积引起的信号运行延迟将超过由于门长度减小所带来的利益。此外，随着电路几何尺寸的缩小，电路的本征延迟也因金属互连线电阻的增加和互连的电容效应而增长。

通过敏感性的测试，我们可以大致了解对于任意结点的矩产生影响的因素有哪些，而对电路中参数的适当的调整，可以控制各点的矩的值，从而达到调整延时等作用，以此来辅助电路设计。

## 5.5 本章小结

本章从电路的电阻、电容以及任意结点的 $j-1$阶矩对电路中某固定结点的第 $j$ 阶矩进行了分析，从试验数据中不难看出，由于计算矩的过程是一个迭代的过程，因此它的敏感度是复杂的。对于电阻 R，对于固定结点 $i$，任一结点 $k$ 对它的 $j$ 阶矩的影响，是沿着结点 $i$ 所在支路，自结点 $i$ 开始，所有序号大于 $i$ 的结点的电容与上一阶矩的乘积的和；对于电容 C，从输入源点至这两者之间的路径的公共部分的电阻和其上一阶矩的乘积的和。而对于任意结点的 $j-1$阶矩，结点 i 的 $j$ 阶矩和结点 k 的 $j-1$阶矩的比值，满足他们至源点的公共部分的 R 和结点 k 所连接电容 $C_k$ 的乘积的和。

# 第六章 总 结

## 6.1 符号化矩量计算

本文主要通过对 BDD 算法和 RICE 算法的研究，提出了结合 BDD 结构和树状结构的分析 RC 树状电路矩的分析方法。算法首先通过反向遍历电路，获得电路中所有电容的树状结构，接着通过正向遍历获得所有除了电容之外，即所有的电阻的 BDD 结构，而这个 BDD 结构的最下面的非地结点（从电路图的角度分析，该结点即最靠近输入源的非地结点）的正边连接着前面反向遍历得到的电容树。

建立完整个电路的结构，在计算各阶电路矩的时候，可以通过不同的目的来实现不同的功能。若想直接通过数值计算得到最后各阶矩的值，可以通过设立数据缓冲区来存储各点前一阶矩的值，避免重复计算。若想分析某阶矩的表达式，或者它对电路中某个元件的敏感性，可以通过遍历符号化算法产生的电路结构，输出该结点这阶矩的表达式。

## 6.2 时序估计

首先经过对一些时序估计公式的研究，当前最为简单精确的近似是 DM1、DM2 和 D2M 近似。前 2 个是由 Kahng 和 Muddu 提出的算法，只要的思想是通过利用前 3 阶矩 $m_0, m_1, m_2$ 求得电路的主极点，再求解时域上的方程来近似匹配延时，当 2 个不同的主极点近似为一个主极点时，就成了 DM2 近似。若再考虑更高阶矩，就可以产生类似的 DM3 等近似。显然的是，随着更高阶矩的加入，精度会越来越高，但复杂度也会急剧增加。因此，综合考虑速度和精度，本文中只讨论了前 2 种近似。通过试验数据发现 DM2 在电路图的中间部分，即不是近端也不是远端，它的近似度比较好。

除了这些利用公式推导得出的近似公式，还有 Alpert 等人提出的一个经验公式 D2M。这个主要建立在试验基础上，通过归纳总结的近似在任意电路图的远端都有较好的近似，但在近端仍与实际值有不小的差距。

在测试速度运行效率方面，SPICE 的测试由于需要通过不停的计算瞬态分析来获得波形图，因此当阶数较大时，运行效率十分低，速度很慢。相对的，利用符号化分析法

去构建一个电路的结构，由于构建过程只需要一次，后面计算的过程只是不停的迭代遍历这个建好的结构，因此速度就相对要快很多。当阶数不大的时候，效果并部明显，而当阶数达到上千甚至上万的时候，就可以明显看到符号化分析的优势所在。另外，从试验数据中可以观察到，符号化分析中时间主要耗费在计算第一阶矩的过程中，即包括读取数据，分析电路结构，建立相对应的树状结构和 BDD 结构等。而在之后计算二阶矩，或者更高阶的矩，由于不需要重新建立结构，只需要迭代运算，因此所耗费的时间就比较少。

总而言之，利用符号化分析方法，可以高效的计算较大规模电路的矩量。当前比较欠缺的就是处理电路的单一性，由于研究工作主要集中在 RC 树状电路这种典型的互连线模型，随着特征尺寸的变小，需要考虑更多高频部分的因素，因此 RLC 模型将会是今后考虑的重点。

## 6.3 矩的敏感性测试

由于符号化分析法的特殊性，可以轻松的得到任一阶矩的表达式，因此通过观察表达式，分析了各个元件与矩量之间的联系。

首先是电阻。对于固定结点$i$，任一结点$k$对它的$j$阶矩的影响，是沿着结点$i$所在支路，自结点$i$开始，所有序号大于$i$的结点的电容与上一阶矩的乘积的和。当两个结点处于不同支路时，那么互相之间是没有影响的。而当这两个结点处于同一条支路上时，那么改变任一个结点，都会对另外一个产生影响，至于两者之间的联系度就如上文提出的，是一个复杂的电容与上一阶矩的乘积的和。因此，电阻和矩之间并没有直接的联系，而前后两阶矩的比值随着电阻的变化几乎是线性增加的。

其次是电容。任意结点的矩的值和任一电容的关系是，从输入源点至这两者之间的路径的公共部分的电阻和其上一阶矩的乘积的和。和电阻不同的是，电阻的公共部分是后向的，即序号大于$\max(i, j)$的部分，而电容的公共部分是前向的，即序号小于$i$和$j$的公共部分。因此，不论这两个电容如何分布，在同一支路或者在不同支路，只要这两者之间有公共部分，那么互相就存在影响。而影响的大小如同电阻分析的一样，电容和矩之间也没有直接的联系，而前后两阶矩和电阻的乘积的和的比值是线性增加的。

最后是任一结点的前一阶矩对矩量的影响。从整个建立结构到计算的过程中，明显的是矩量的计算是一个迭代计算的过程，每次得到新一阶的矩都需要上一阶所有结点矩的值。因此，对于任意结点$i$的$j$阶矩和结点 k 的$j-1$阶矩的比值，满足他们至源点的公共部分的 R 和结点 k 所连接电容$C_k$的乘积的和。由于每次迭代运算矩的时候，就如同

分一个直流电路，而此时电容所代表的元件就是值为电容值和上一阶矩的乘积的电流源，因此结点 k 的 $j-1$阶矩的直接作用对象就是电容$C_k$，因此和电阻电容不同的是，对于任一结点的前一阶矩，它的公共部分的取值只有电阻 R 部分，而电容$C_k$是固定的值。若通过改变公共部分以外的电阻和电容来改变结点 k 的 $j-1$阶矩，那么前后两阶矩之间的关系就应该成线性关系。反之，若由于改表了公共部分的电阻值而影响了结点 k 的 $j-1$阶矩，那么这前后两阶矩之间的关系就较为复杂。

## 6.4 本章小结

本章主要简单的总结了在本文中提出的新的符号化计算矩量的方法，另外通过分析计算电路的精度和速度对该分析法与 SPICE 仿真进行了比较。而对于矩量敏感性的测试则说明了矩量对于电路元件各参数的依赖性。

# 参 考 文 献

[1] L. Nagel, “SPICE2, A computer program to simulation semiconductor circuits,” Univ. California, Berkeley, CA, TR ERL-M520, May 1995

[2] P. Feldmann and R. W. Freund. “Reduced-order modeling of large linear sub circuits via a block Lanczos algorithm”, In Proc. 32nd ACM/IEEE Design Automation Conference, pp 474-479, New York, New York, 1995

[3] L. T. Pillage and R. A. Rohrer, “Asymptotic waveform evaluation for timing analysis,” IEEE Trans. Computer-Aided Design, vol. 9, pp. 352-366, Apr. 1990

[4] K. Brace, R. Rudell, and R. Bryant, "Efficient implementation of a BDD package," in Proc. 27th IEEE/ACM Design Automation Conference, pp. 40-45, June, 1990.

[5] P. Feldman and R. W. Freund, “Efficient Linear Circuit Analysis by Pade Approximation Via the Lanczos Process”, IEEE Trans. Computer-Aided Design, vol. 14, no. 5, pp. 639-649, May 1995

[6] L.M.Silveira, M.Kamon, I.Elfadel, and J.White, “A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of arbitrary RLC circuits,” in IEEE/ACM pro. ICCAD, Nov. 1996, pp. 88-294.

[7] K. J. Kerns, I. L. Wemple, A. T. Yang, “Stable and Efficient Reduction of Substrate Model Networks using Congruence Transformation”, ICCAD, Nov. 1995

[8] A.Odabasioglu, M.Celik, and L.T.Pillage, “PRIMA: Passive reduced-order interconnect macro modeling algorithm,” IEEE Trans. Computer-Aided Design, vol. 17, No. 8, pp. 645-654, Aug.1998.

[9] S. Minato, "Zero-suppressed BDD's for set manipulation in combinatorial problems," in Proc. 30th IEEE/ACM Design Automation Conf., (Dallas, TX), pp.272-277, 1993.

[10] Roland W.Freund, “Krylov-subspace methods for reduced-order modeling in circuit

simulation”, http://cm.bell-labs.com/cs/doc/99.

[11] W. Sansen, G. Gielen, and H. Walscharts, "A symbolic simulator for analog circuits," in Proc. ISSCC, 1989, pp. 204-205.

[12] G. Gielen, H. Walscharts, and W. Sansen, "ISAAC: A symbolic simulator for analog integrated circuits," IEEE J. Solid-State Circuits, vol. 24, no. 6, pp. 1587-1597, December. 1989.

[13] F. Femandez, A. Rodriguez-Vazquez, and J. Huertas, "A tool for symbolic analysis of analog integrated circuits including pole/zero extraction," in Proc. ECCTD, 1991, pp.752-761.

[14] J. Kluwer "Interactive ac modeling and characterization of analog circuits via symbolic analysis,". Analog Integrated Circuits and Signal Process., vol. l, pp. 183- 208, November, 1991. [15] S. Seda, M. Degrauwe, and W. Fichtner, "A symbolic analysis tool for analog circuit design automation," in Proc. ICCAD, 1988, pp. 488-491.

[16] K. Glover, “All optimal Hankel-norm approximation of linear multivariable systems and their r L∞ –error bounds.” Int. J. Control, Vol.39, No.6, pp 1115-1193, 1984.

[17] Jing-Rebecca Li, etc. “An efficient lyapunov equation-based approach for generating reduced-order models of interconnect,” Proceedings of the 6th ACM/IEEE Design Automation Conference, pp.1-6, June 1999.

[18] Janet M.Wang, Qingjian Yu and Ernet S. Kuh, “Passive model order reduction algorithm based on Chebyshev expansion of impulse response,” Proceedings of the 6th ACM/IEEE Design Automation Conference, pp. 520-526, June 2000.

[19] "Lazy-expansion symbolic expression approximation in SYNAP," in Proc. ICCAD, 1992, pp. 31C317.

[20] G. Wierzba et al., "SSPICE-A symbolic SPICE program for linear active circuits," in Proc. Midwest Symp. on Circuits and Systems, pp. 1197-1201 vol 2, 1989.

[21] A. Konczykowska and M. Bon, "Automated design software for switched-capacitor IC's with symbolic simulator SCYMBAL," in Proc. DAC, 1988, pp. 363-368.

[22] M. Hassoun and P. Lin, "A new network approach to symbolic simulation of large-scale networks," in Proc. ISCAS, 1989, pp. 806-809.

[23] Kevin J. Kerns and Andrew T. Yang, “Stable and Efficient Reduction of Large, Multiport RC Networks by Pole Analysis via Congruence Transformations.” IEEE Trans. Computer-Aided Design, vol. 16, No. 7, pp.734-744, July 1997.

[24] L. Huelsman, "Personal computer symbolic analysis programs for undergraduate engineering courses," in Proc. ISCAS, 1989, pp. 798-801.

[25] Curtis L. Ratzlaff and Lawrence T. Pillage, “RICE: Rapid Interconnect Circuit Evaluation Using AWE”, IEEE Trans. Computer-Aided Design, vol. 13, No. 6, pp 763-776, June 1994

[26] R. W. Freund and P. Feldmann. “Reduced-order modeling of large passive linear circuits by means of the SyPVL algorithm,” In Tech. Dig. 1996 IEEE/ACM International Conference on Computer-Aided Design, pp. 280-287, Los Alamitos, California, 1996. IEEE Computer Society Press

[27] R. W. Freund and P. Feldmann. “The SyMPVL algorithm and its applications to interconnect simulation,” In Proc. 1997 International Conference on Simulation of Semiconductor Processes and Devices pp 113-116, Piscataway, New Jersey, 1997. IEEE

[28] James D. Ma and Rob A. Rutenbar. “Fast Interval-Valued Statistical Modeling of Interconnect and Effective Capacitance,” IEEE trans. Computer-Aided Design, vol. 25, NO. 4, APRIL 2006, pp. 710-724

[29] Y. Liu, L. T. Pileggi, and A. J. Strojwas, “Model order-reduction of RC(L) interconnect including variational analysis, “ in Proc. Design Automation Conf., New Orleans, LA, 1999, pp. 201-206.

# 附录一 符号与标记

[1]MOR, Model Order Reduction 的缩写，模型降阶

[2]AWE 算法，Asymptotic Waveform Evaluation 的缩写，渐进波形求值算法

[3]BDD，Binary Decision Diagram 的缩写，二叉判定图算法

[4]OBDD，Ordered Binary Decision Diagram，有序二叉判定图

[5]ROBDD，Reduced Ordered Binary Decision Diagram，最简有序二叉判定图。当前通常意义上提到的 BDD 算法就是 ROBDD 算法。

[6]矩匹配，moment matching

[7]RICE，Rapid Interconnect Circuit Evaluation 的缩写，提供了快速计算电路矩的方法

[8]Elmore Delay，Elmore 延时，电路中一阶矩正好等于 Elmore 延时

[9]DM1，DM2，Kahng 和 Muddu 提出的近似延时的方法，英文全称分别为 Delay Metric 1 和 Delay Metric 2。

[10]D2M，Alpert 提出的近似延时的经验公式，英文全称为 Delay via two moments。

[11] $m_i^j$ 代表第 $i$ 个结点的第 $j$ 阶矩

# 攻读硕士学位期间已发表或录用的论文

[1] 徐迪，“树状 RC 电路的符号化矩量计算”，上海交通大学学报

# 攻读硕士学位期间参与的科研项目

[1] 浦江人才计划，“纳米集成电路物理设计有关辅助工具研究”

# 致 谢

经过数月的努力，硕士阶段的论文终于结束了。在这里，最要感谢的是我的导师，施国勇教授。自从本科阶段进入施老师的团队之后，在施老师的悉心教导下，我从对 EDA 领域的茫然不知，到略微了解，到如今的略有突破。施老师严谨的教学态度，开明的引导方法都让我获益匪浅，在此，我谨向施老师表达我最深深的谢意。

感谢胡薇薇老师、谢憬老师在平时的学习、生活中给予的关心和鼓励。

感谢郝志刚师兄在我初涉科研时给予的大力帮助和指导。感谢本课题组所有成员共同营造了浓厚的学术氛围和舒适的环境。感谢于雪红、孟晓旋、刘安等同学在研究过程中的互相帮助，互相学习。这样的经历是本人终生难忘的。在此尤其要感谢的是已经毕业的陈薇薇师姐，正是她的巨大贡献给我竖立了良好的榜样。

B0621091 是一个温暖的大家庭，感谢班长胡泊同学、党支书张朝华同学，你们在生活学习等各方面的工作让我体会到了家的温暖气氛，也使之成为本人一生中宝贵的回忆。也感谢所有同班的、同院的同学，是你们让我的生活变的充实。

感谢同寝室的郑马骏同学，你的天马行空的思维让我开阔了眼界。和你相处的时间让我学到了很多原本不知道的东西。本人十分珍惜这段共同学习、共同生活的日子。

感谢帮助过我、关心过我的朋友，你们的关心让我充满动力。

最后衷心的感谢我的父母在我十多年学习生涯中对我最无私的理解和支持，没有你们的帮助，我也达不到如今的高度。希望本文的完成能让所有关心我的家人朋友感到欣慰。