

上海交通大学硕士学位论文

振荡电路周期主极点符号化计算方法及其在电路优化中的应用¹

硕士研究生：朱彦

学号：1102109024

导师：施国勇教授

专业：集成电路工程

所在单位：微电子学院

答辩日期：2012年12月

授予学位单位：上海交通大学

¹ 本研究由国家自然科学基金（项号 61176129）资助

Dissertation Submitted to Shanghai Jiao Tong University
for the Degree of Master

**SYMBOLIC CALCULATION OF
OSCILLATOR PERIODIC MAIN
ROOT-LOCUS AND ITS APPLICATION
IN CIRCUIT OPTIMIZATION**

Candidate:	Yan Zhu
Student ID:	1102109024
Supervisor:	Prof. Guoyong Shi
Speciality:	Integrated Circuit Engineering
Affiliation:	School of Microelectronics
Date of Defence:	Dec, 2012
Degree-Conferring-Institution:	Shanghai Jiao Tong University

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《振荡电路周期主极点符号化计算方法及其在电路优化中的应用》，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：朱彦

日期：2013年2月20日

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：宋彦

指导教师签名：施国勇

日期：2013年2月20日

日期：2013年2月20日

振荡电路周期主极点符号化计算方法及其在电路优化中的应用

摘要

对振荡器电路做周期性稳态分析，并在其基础上对电路进行线性化，可以得到周期性变化的小信号模型。对这些小信号模型求根，可以得到复平面上的根轨迹。这一信息能够直观地解释振荡器工作的特点，为设计者带来有用的指导信息。

当前，已经有数值化的分析方法可以求得这一根轨迹。然而，这一分析方法包含了重复、冗余的计算，能够提供的信息有限。

本文提出了一种基于符号化分析的高频振荡电路主极点根轨迹求解方法，利用符号化分析方法的优点，结合多项式计算的特征，为运算带来更高的效率。同时，这一计算方法能够提供根轨迹敏感度信息，为设计带来更多有用的指导。

然而，利用符号化方法进行近似电路的拓扑结构比较时，其符号化表示需要进行重构，这给不同电路的结构分析带来了不便。本文提出了一种在无需重构的前提下，对电路元件进行动态插入与删除的方法。这一方法能够为电路分析带来更多的便利。

关键词（四号黑体）： 符号化分析、高频振荡电路、根轨迹、符号化结构动态变化

SYMBOLIC CALCULATION OF OSCILLATOR PERIODIC MAIN ROOT-LOCUS AND ITS APPLICATION IN CIRCUIT OPTIMIZATION

ABSTRACT

Linearizing an oscillator around its periodic steady state (PSS) would generate a small-signal model with time-varying parameter value. The time-varying root-locus of this model can be plotted on the complex plane. This information explains how oscillator works during its PSS, which is of benefit to designers.

Currently, there's a numerical method to generate this root-locus, yet the calculation process is repeated and inefficient. Also, this method provides limited information.

In this paper, a symbolic main root-locus calculation method is proposed, which utilizes the advantage of both symbolic method and polynomial root calculation methods to advance the calculation efficiency. Sensitivity information can be generated with ease with this method, which can provide further information to help with design.

To lend more flexibility to symbolic method for comparison of similar circuit topologies, a methodology to insert and delete devices without reconstruction of the data structure is proposed. Applying the method to structural comparison of different circuits would bring about more efficiency.

KEY WORDS: symbolic method, high-frequency oscillators, root-locus, dynamic modification of symbolic data structure

目 录

第一章 绪论	1
1.1 模拟、射频电路仿真方法的研究背景	1
1.2 模拟电路的两类分析方法	1
1.3 射频电路的周期性时变线性系统模型	2
1.4 振荡器电路的时变根轨迹分析方法	4
1.5 数值化的振荡器主极点根轨迹计算方法	8
1.6 本文主要内容与章节安排	10
第二章 符号化仿真方法简介	12
2.1 符号化仿真方法总体介绍	12
2.2 二分决策图	12
2.3 GPDD 仿真算法基本原理	15
2.3.1 基本约束条件	15
2.3.2 有向图转换规则	16
2.3.3 GPDD 构造规则	17
2.3.4 GPDD 的 s 展开	22
2.4 本章小结	23
第三章 符号化振荡电路时变主极点根轨迹分析方法	24
3.1 总体流程	24
3.2 大信号部分分析流程	25
3.2.1 PSS 仿真条件	25
3.2.2 线性化电路	25
3.3 小信号部分分析流程	27
3.3.1 数据结构的建立	27
3.3.2 周期时变特征多项式系数求解	27
3.3.3 连续 Muller 主极点求解方法	28
3.4 根轨迹敏感度分析方法	32
3.5 本章小结	34
第四章 符号化电路综合方法	36
4.1 问题的提出	36

4.2 元件极限操作	36
4.3 元件添加操作	44
4.4 本章小结	52
第五章 实验结果分析	53
5.1 符号化仿真器与数值化仿真器的对比分析	53
5.1.1 测试电路 1: 交叉耦合振荡器	54
5.1.2 测试电路 2: 带偏置电感的交叉耦合振荡器	56
5.1.3 符号化高频电路主极点根轨迹算法的效率分析	58
5.2 主极点根轨迹敏感度分析结果	61
5.3 本章小结	67
第六章 结束语	68
6.1 主要工作与创新点	68
6.2 后续研究工作	68
参 考 文 献	70
致 谢	73
攻读硕士学位期间已发表或录用的论文	75

图 录

图 1-1 一个简单的 COLPITTS 振荡器.....	5
图 1-2 COLPITTS 振荡器的时变根轨迹	5
图 1-3 COLPITTS 振荡器主极点实数部分与输出电压 PSS 解的对应关系.....	6
图 1-4 COLPITTS 振荡器主极点根轨迹及相位噪声随 Q 值改变而发生的变化.....	7
图 1-5 电感 L 的改进节点法矩阵表示	8
图 1-6 数值化振荡器主极点根轨迹计算过程示意图	9
图 2-1 示例真值表.....	13
图 2-2 示例布尔表达式的 BDD 和 ROBDD	13
图 2-3 不同顺序的 ROBDD 大小比较	14
图 2-4 一个简单的 RC 电路及其有向图.....	17
图 2-5 RC 电路有向图的左图和右图.....	18
图 2-6 同构图检测.....	19
图 2-7 RC 电路对应的 GPDD.....	20
图 2-8 GPDD 求值过程.....	21
图 2-9 去除输入输出后的 RC 电路及其有向图.....	22
图 2-10 RC 电路特征多项式的 GPDD 结构及 s 展开 BDD 结构	23
图 3-1 符号化振荡电路主极点根轨迹分析流程	24
图 3-2 DC-OP 工作点分析示意图	25
图 3-3 MOS 管完整准静态模型 ^[28]	26
图 3-4 无源 RC 时变电路及对应的 s 展开 BDD 结构	28
图 3-5 MULLER 迭代过程示意图	29
图 3-6 连续 MULLER 主极点求解过程示意图	31
图 4-1 电阻、电容、电感进行短路、断开操作对应的元件值及导纳值	37
图 4-2 一个 GPDD 例子	38
图 4-3 P 取极限值后的 GPDD 结构.....	39
图 4-4 应用极限算法的一个例子	41
图 4-5 极限运算的数据结构.....	42
图 4-6 撕裂节点并接入元件示意图	45
图 4-7 接入元件连接不同节点示意图	45
图 4-8 不包含 P 的原电路 GPDD 结构	47
图 4-9 添加 P 后的 GPDD 结构	47
图 4-10 调整后的包含 P 的电路的 GPDD 结构.....	48
图 4-11 元件 P 由无穷大变化到具体值的 GPDD 变化过程	49
图 4-12 元件 P 由 0 变化到具体值的 GPDD 变化过程.....	50
图 5-1 交叉耦合振荡器.....	54
图 5-2 交叉耦合振荡器的主极点根轨迹	55
图 5-3 交叉耦合振荡器主极点根轨迹实数部分与 PSS 分析结果对比图.....	55
图 5-4 带偏置电感的交叉耦合振荡器	56
图 5-5 带偏置电感的交叉耦合振荡器的主极点根轨迹	57
图 5-6 带偏置电感的交叉耦合振荡器主极点根轨迹实数部分与 PSS 分析结果对比图.....	58
图 5-7 调整偏置电感对振荡器 2 的性能影响	60
图 5-8 根轨迹变化对 C_0 的敏感度.....	61
图 5-9 根轨迹变化对 L_{BOT} 的敏感度.....	62
图 5-10 根轨迹变化对 L_{TOP} 的敏感度.....	63

图 5-11 对左极值点影响最大的参数示意图	64
图 5-12 对右极值点影响最大的参数示意图	65

表 录

表 2-1 GPDD 图约化规则	18
表 5-1 交叉耦合振荡器的各项参数值	54
表 5-2 带偏置电感的交叉耦合振荡器的各项参数值	57
表 5-3 符号化分析方法与 QZ 数值化分析方法的效率比较	59
表 5-4 符号化方法的细节统计	59
表 5-5 对根轨迹左极值点最敏感的电路元件	64
表 5-6 对根轨迹右极值点最敏感的电路元件	65
表 5-7 基于两种方法的 Osc-1 系数计算时间比较	66
表 5-8 元件添加操作细节统计	67

第一章 绪论

1.1 模拟、射频电路仿真方法的研究背景

随着大规模集成电路技术的不断发展，片上系统(System on Chip, SoC)的分析、设计及仿真技术也日趋完善。众所周知，一般的片上系统分为数字电路及模拟、射频电路两部分。计算机辅助设计(CAD)工具及仿真算法带来了数字电路的飞速发展，人们只需要在顶层通过硬件语言描述电路，并制定一系列综合、布局布线等限制，便可依靠工具强大的自动计算、仿真、布局布线功能，得到最终所需要的电路版图文件。这一点使得数字电路的开发周期大大缩短。然而，模拟、射频电路受益于CAD工具的程度却远远不及数字电路深。这是由于模拟、射频电路设计的复杂性、经验性等造成的。数字电路的设计往往与逻辑相关，而模拟、视频电路的设计却需要符合更多复杂的指标。一个模拟、射频电路从结构设计、各个元件尺寸的合理调整、仿真分析到最终的版图设计、布局布线，无不需要工程师们长期积淀经验与智慧。

由于流片过程成本高、周期长，因此电路设计对于仿真器的要求非常高。一个准确、高效的仿真器将对设计周期的缩短带来本质的影响。随着电路规模的日益增大，仿真器的速度及准确性都受到了考验。例如，若在现代仿真器上对一个完整的射频接收器系统作精确分析，可能需要花费几天的时间。若使用简化模型，则时间大大缩短，但是其结果的准确性则受到了限制。即使前端设计的仿真结果显示电路系统性能良好，后端设计的细微差异也可能极大地影响最终的电路性能。因此，要设计出符合指标的电路，需要工程师们反复调整参数，并运行仿真工具进行结果验证。日益缩短的设计周期要求，与不断增大的电路规模之间的矛盾，需要通过更高效、更准确的模拟、射频电路仿真方法来调和。

1.2 模拟电路的两类分析方法

当前，分析模拟电路的仿真方法主要可以分为两类：数值化分析方法及模拟化分析方法。数值化分析方法将电路表示为矩阵形式，并通过一系列求解矩阵的操作来得到电路的解。而符号化方法将电路以符号形式存储在数据结构中。当电路的拓扑结构未发生改变时，数值化方法的矩阵需要重新生成；而符号化方法的数据结构则无需更改，只需调整对应的元件的具体参数即可。

由于数值化分析方法将电路表示为矩阵形式求解，因此其对应的数据结构相对简单。得益于数值分析理论的发展，数值化方法的仿真速度比较快。加州大学伯克利分校(University of California, Berkeley)在上世纪 70 年代发展起来的数值电路 SPICE 仿真器，成为了当今工业界流行的仿真器(如 Synopsys Hspice, Cadence Spectre, Agilent ADS 等)的基础。它可以分析包含场效应管、二极管等一系列非线性元件的复杂电路，得到电路的直流工作点、交流特性及传输函数的频域特性等。数值化分析方法擅长于以具体数值表征电路的性能，因此非常适用于验证电路是否符合设计的要求。

然而，数值化分析方法无法进一步提供电路的信息，如哪个元件对电路的某项性能起决定作用等。传统的设计方法通常是通过手工推算与数值化分析方法相结合，通过手工推算的理论对电路元件加以调整，并不断地通过数值化分析进行验证。然而，手工推算通常基于简化的电路模型，并且在推算的过程中常常需要忽略一些不重要的项，因此无法对电路的性能提供精确的指导信息。对于初学者而言，就只能通过不停的尝试及反复的仿真，才能对主要元件的作用得到一个经验性的认识。这往往需要耗费大量的时间。

符号化分析方法则能够很好地弥补数值化分析方法的这点不足。不同于数值化分析方法的矩阵分析过程，符号化分析方法将电路的时间、频率、元件参数等视作一系列的变量，并构建以这些变量为基础的数据结构，得到符号化的公式，从而直观地解释每个电路元件对电路性能的影响，从而能够反过来知道设计的方向。同时，符号化分析方法避免了数值计算过程中的舍入误差及计算收敛性问题，能够为仿真结果的精度带来更大的可靠性。在调整电路元件参数后，符号化结构无需更改，只需调整对应符号的具体数值，直接计算便可更新符号化公式的具体数值。

然而，符号化分析方法也有其缺陷。符号化表达式由一系列符号化生成项组成，而这些生成项的数目随着电路元件数目的增加而呈指数增长的趋势。这使得符号化方法所能分析的电路尺寸受到了限制。然而由于符号化分析方法能够直观地揭示电路信息，因此在仿真小规模或中等规模的电路时，符号化分析方法仍然具有很好的表现。

1.3 射频电路的周期性时变线性系统模型

我们知道，对模拟电路进行分析时，首先需要通过 DC 仿真得到电路的直流工作点，然后对电路在 DC 工作点附近进行线性化得到小信号电路，最后对电路

进行小信号分析。

与模拟电路的分析相似，分析射频电路时，也是遵循同样的步骤，先对电路进行大信号分析确定工作点，然后再进行小信号分析。然而，与模拟电路不同的是，射频电路往往并不具有一个恒定不变的直流工作点，而具有一个周期性的时变工作点。因此，模拟电路的线性时不变系统模型对于射频电路的分析往往并不适用。[2]综述了一系列经典的射频电路分析方法。[3]介绍了当前主流的射频电路仿真工具的算法及性能。

Zadeh 在 1950 年提出了时变系统函数的理论^[1]，推出了线性时变电路的时变系统函数表示方法。我们可以根据这个理论对具有周期性时变工作点的射频电路进行分析。

对含有储能元件(如电感、电容等)的电路来说，电路的 MNA(Modified Nodal Analysis)方程是一个 N 维向量方程组，表示成如下的形式：

$$\frac{d}{dt}q(v(t)) + i(v(t)) + u(t) = 0, \quad v(0) = v_0 \quad (1-1)$$

其中，公式中的项均为 N 维度向量。 $u(t)$ 表示输入向量， $v(t)$ 包含了所有节点的电压，并可能包含某些支路的电流。对于振荡器来说， $u(t)$ 的值为 0，只要给予适当的初始条件便能够起振。

假设公式(1-1)中的电路有周期性稳态解 $v_{ss}(t)$ ，且周期为 T ，则有 $v_{ss}(t+T) = v_{ss}(t)$ 。 $v_{ss}(t)$ 符合：

$$\frac{d}{dt}q(v_{ss}(t)) + i(v_{ss}(t)) = 0 \quad (1-2)$$

假设周期性稳态解中加入了一个小的扰动，即 $v(t) = v_{ss}(t) + \delta v(t)$ ，则 (1-1) 可以转化为：

$$\frac{d}{dt}q(v_{ss}(t) + \delta v(t)) + i(v_{ss}(t) + \delta v(t)) = 0 \quad (1-3)$$

对式 (1-3) 做一阶 Taylor 展开，并减去 (1-2) 式，我们有

$$\frac{d}{dt}[C(t)\delta v(t)] + G(t)\delta v(t) = 0 \quad (1-4)$$

其中， $C(t) = \left. \frac{\partial q(v(t))}{\partial v} \right|_{v=v_{ss}(t)}$ ， $G(t) = \left. \frac{\partial i(v(t))}{\partial v} \right|_{v=v_{ss}(t)}$ ，两者均为 N 阶矩阵。基于

“小信号分析的频率远高于电路周期性稳态工作点的频率”的假设，我们忽略公式 (1-4) 左边的 $(\frac{d}{dt}C(t))\delta v(t)$ 这一项，可以得到

$$C(t) \frac{d}{dt} \delta v(t) + G(t) \delta v(t) = 0, \quad \delta v(0) = \delta v_0 \quad (1-5)$$

根据 Zadech 的理论^[1]对上式做时变的 Laplace 变换, 得到

$$[C(t)s + G(t)]\delta V(s) = C(t)\delta v_0 \quad (1-6)$$

其中 $\delta V(s)$ 是 $\delta v(t)$ 的 Laplace 变换形式。这一小信号模型在频域中的极点满足下面的式子

$$D(s, t) := \det |sC(t) + G(t)| = 0 \quad (1-7)$$

其中, $D(s, t)$ 表示振荡器的时变特征多项式。

由于我们是在 PSS 解附近对电路进行频域转换与线性化, 因此 $C(t)$ 和 $G(t)$ 都是周期性时变的矩阵。很显然, $D(s, t)$ 是周期性时变的多项式, 因此满足方程的解 s 必然也是周期性时变的。如果我们对 PSS 进行采样, 并且从采样值中求解方程的根 s , 我们就在复平面上得到一个时变根轨迹图。假设 $D(s, t)$ 为 n 阶多项式, 则这个时变根轨迹图将包含 n 个环, 分别表示 n 个解在一个 PSS 周期内扫过的轨迹。这个图可以为电路分析提供许多有用的信息。

1.4 振荡器电路的时变根轨迹分析方法

根轨迹技术是控制理论中的经典系统分析方法, 由 W. R. Evans 在[4]中首次提出。根轨迹技术能够快速地勾画出在某些环路传输参数(如开环增益)变化时, 闭环控制系统的极点将如何移动。控制系统的极点在复平面上的分布与系统的稳定性及过渡过程的性能有密切关系。

现代振荡器电路的发展要求仿真器能够快速、准确地分析其性能。在一些列振荡器分析方法中, 根轨迹技术显示了其独特的价值。根轨迹分析方法被广泛地用于分析振荡器的起振[5][6]、多谐振荡器在两个暂稳态之间来回震荡的过程[7]以及非线性系统的稳定性[8]。

[9]首次提出了振荡器电路的时变根轨迹(Time-Varying Root-Locus, TVRL)分析方法, 这一方法很好地结合了 1.3 节中提到的射频电路的时变系统理论及根轨迹理论, 通过一系列的 CAD 技术画出了 LC 振荡器处于稳态过程时, 在一个周期内的根轨迹示意图, 并根据不同的根轨迹分布, 能够有效地分析 LC 振荡器的性能。

我们用一个简单的 Colpitts Oscillator^[10]来解释这一过程。

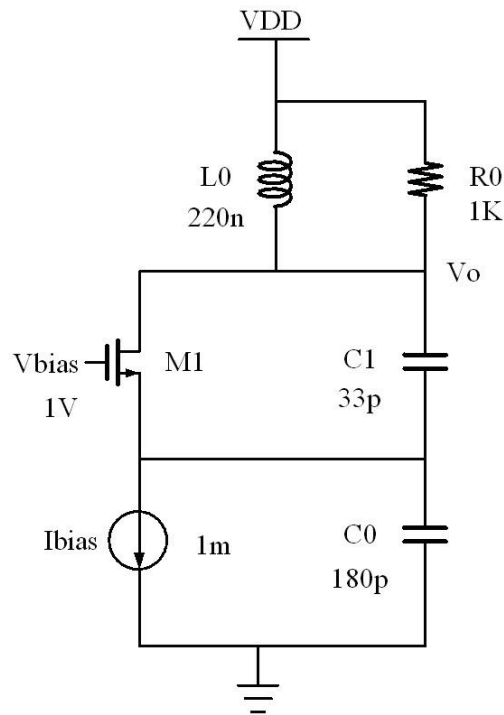


图 1-1 一个简单的 Colpitts 振荡器
Figure 1-1 A simple Colpitts Oscillator

这个 Colpitts 振荡器中的电感、电容均为理想原件，其有限 Q 值用 R0 来模拟。用 1.3 的理论获得的时变根轨迹如下图所示。

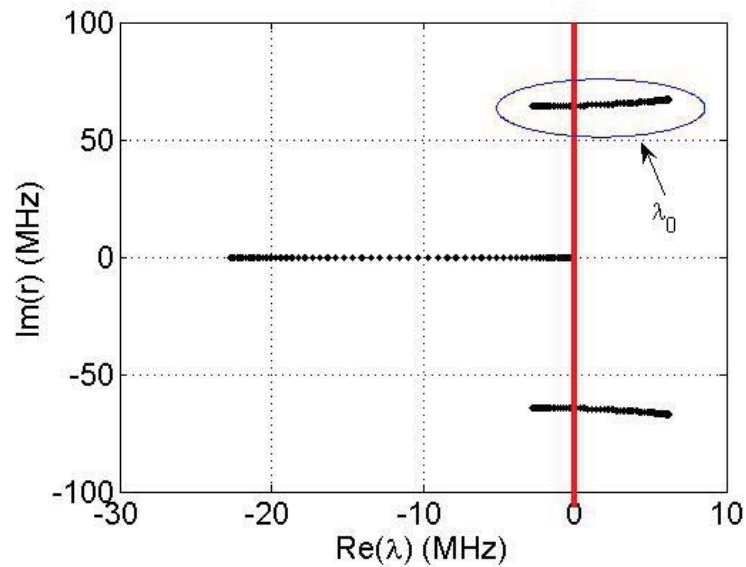


图 1-2 Colpitts 振荡器的时变根轨迹
Figure 1-2 Time-varying root-locus of the Colpitts Oscillator

可以看到，图中有一对共轭的轨迹穿越虚轴，而另一个实根轨迹则保持在左半平面上。这对共轭的轨迹就是我们分析的主极点根轨迹，包含了振荡器的主要特性。由于其共轭性，我们只分析虚部为正的轨迹 λ_0 。它有如下的一些特征：

(1) λ_0 的虚数部分围绕着 Colpitts Oscillator 的频率 60MHz 进行振荡。

(2) λ_0 的实数部分为极负值时，振荡器处于截止状态。实数部分为极正值时，振荡器处于饱和状态。穿越虚轴时，振荡器处于亚阈值状态。为了说明这个现象，我们将的 λ_0 实数与时间的对应关系画出，并与输出电压 V_o 的 PSS 解进行对照。

结果如图 1-3 所示。

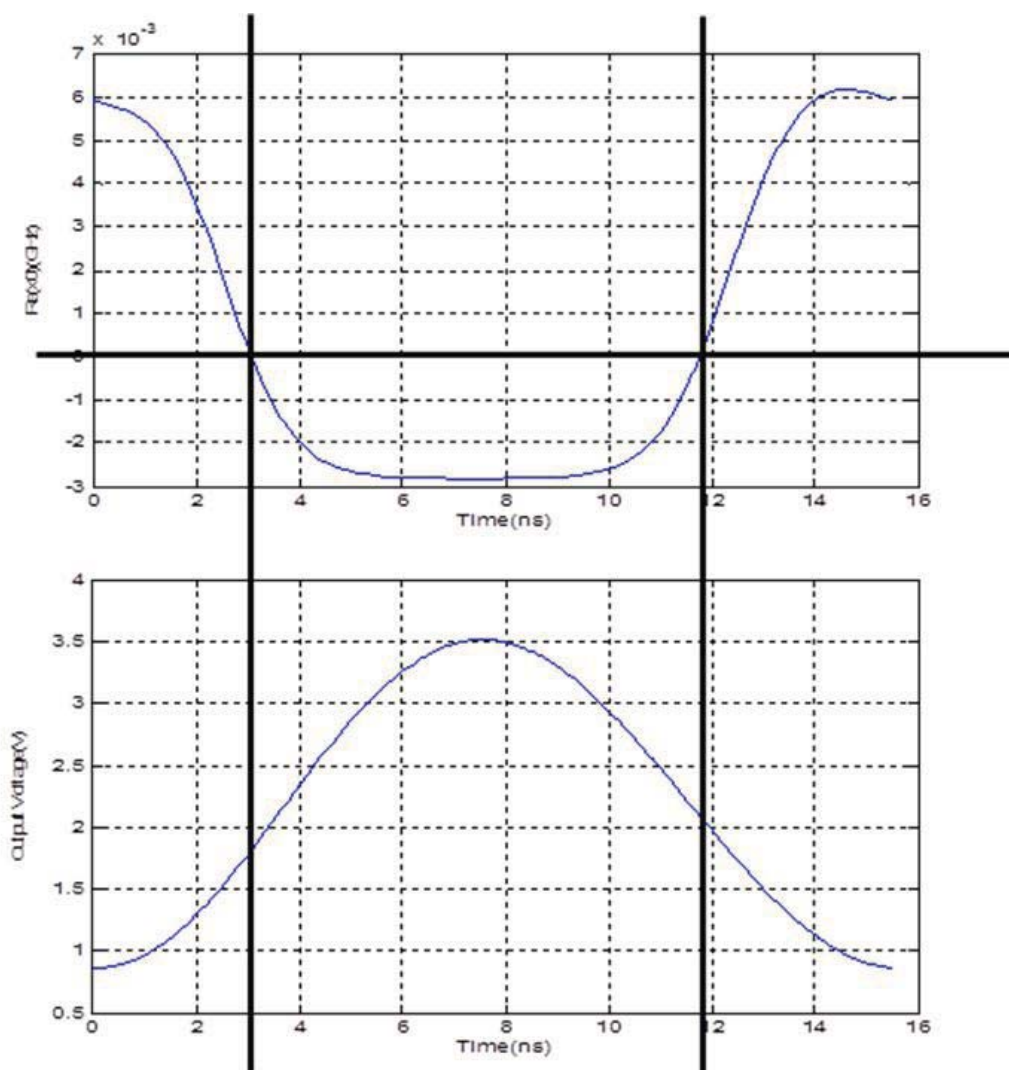
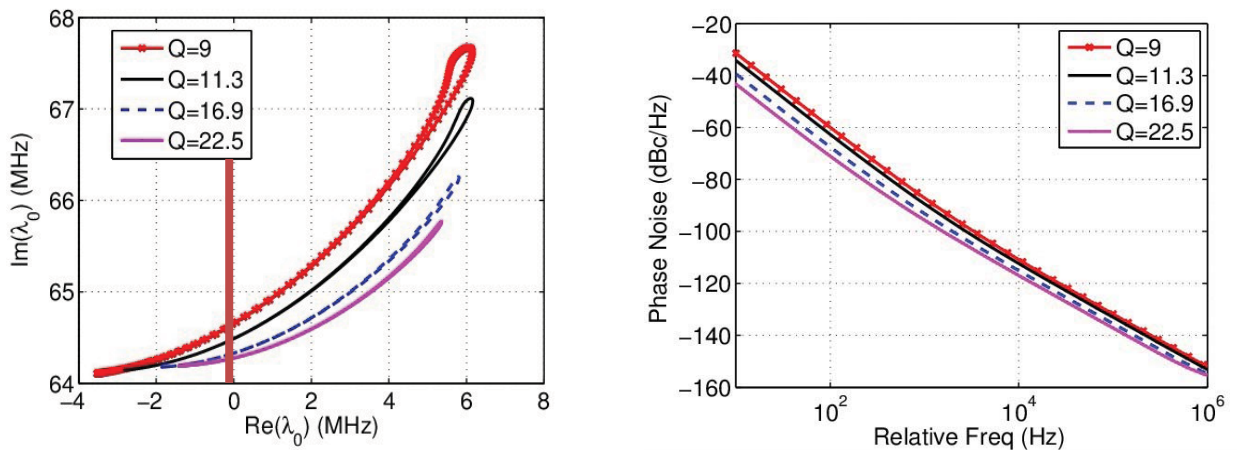


图 1-3 Colpitts 振荡器主极点实数部分与输出电压 PSS 解的对应关系

Figure 1-3 The relationship between the real part of Colpitts Oscillator's main root and the PSS solution of its output voltage

从图 1-3 中可以看出, λ_0 的实数部分为最大值时, 输出电压 V_o 达到最小值, 振荡器处于饱和状态, 振荡器表现为负阻抗, 使 Q 值变高。当 λ_0 的实数部分为最小值时, 输出电压 V_o 达到最大值, 振荡器截止, 表现为正阻抗, 使 Q 值变小。

为了更好地了解时变根轨迹的指导作用, 我们做一个实验, 改变图 1-1 中 R_0 的值, 使得振荡器 LC tank 的 Q 值从小变大。同时, 我们更新 I_{bias} 的值, 使得输出电压的幅度保持不变。得到的根轨迹如图 1-4a) 所示。



a) Colpitts 振荡器主极点随 Q 值变化而发生的变化

a) The variation of the main time-varying root-locus of the Colpitts Oscillator with regard to different tank Q value

b) Colpitts 振荡器相位噪声随 Q 值变化而发生的变化

b) The variation of the phase noise of the Colpitts Oscillator with regard to different tank Q value

图 1-4 Colpitts 振荡器主极点根轨迹及相位噪声随 Q 值改变而发生的变化

Figure 1-4 The variation of the main time-varying root-locus and phase noise performance of Colpitts Oscillator with regard to different tank Q value

在调节的过程中, 我们发现, 随着 Q 的增加, R_0 值越来越大, 而我们需要更少的输入电流 I_{bias} 就能够保持同样的输出幅度。反映到图 1-4a) 中的时变根轨迹图中来, 我们可以看到, 轨迹的极左、极右值均不断向虚轴靠近。这表示系统在震荡的过程中, 随着 Q 值的增大, MOS 管为系统贡献增加消耗的正导纳 (当轨迹处于左半平面时) 受到了限制, 无法深入到左半平面。同时, 补偿损耗的负导纳 (当轨迹处于右半平面时) 亦因为输入电流 I_{bias} 的减少而有所减少, 表现出整体功耗的下降。这个图像还可以给出一个定性的结论, 即随 Q 值的增大, MOS 管作为正导纳所导入的噪声以及作为负导纳所导入的噪声均有所减少, 能够表现出更优的相噪性能。这一结论在图 1-4b) 中得到了验证。

从上面的分析中, 我们可以看出, 对振荡器的主极点根轨迹分析, 能够使设

计者对振荡器工作的过程以及性能有更加形象化的了解。在这里，我们只给出一些定性的分析，表明振荡器主极点根轨迹的作用。除了能够对振荡器电路自身作出分析外，主极点根轨迹对不同电路结构的分析，也能带来很大的便利。更加详细的分析可参考文献[21]。在此不再赘述。

1.5 数值化的振荡器主极点根轨迹计算方法

我们已经知道，振荡器主极点根轨迹能够给电路的分析带来新的便利。因此，我们需要探寻一种高效的计算方法，使得仿真过程能够更加快捷。同时，我们希望在计算过程中，得到尽可能多的指导信息。

求根轨迹，必然需要通过不同的小信号电路，对(1-7)中的系统时变特征方程进行求根运算。[10][11]总结了当前主流的系统方程求根算法。这些算法主要可以分为两大类。一类是将系统方程转化为如公式(1-7)中所示的 $C(t)$ 和 $G(t)$ 矩阵，并通过改进分解(Modified Decomposition, MD)法或者QZ方法求解电路方程的根；另一类是将系统方程表示为具体的多项式，继而求解多项式的根。[9]中使用了第一类基于矩阵数值的QZ方法，通过对 $C(t)$ 和 $G(t)$ 矩阵的运算得到结果。

我们注意到，LC振荡器电路中含有电感，转化到频域后，电感值是 $\frac{1}{sL}$ ，如果直接代入 $C(t)$ 矩阵，则 s 将出现在分母中，不利于求解。有两种简单的方法能够解决这个问题。第一种是改进节点法(Modified Nodal Analysis)分析^[22]，为矩阵扩展一行一列。电感元件在矩阵中的表示如图1-5所示。

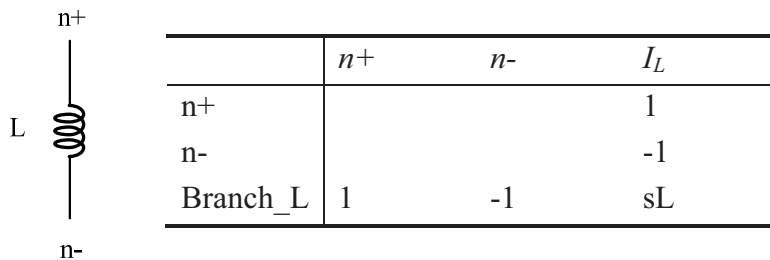


图 1-5 电感 L 的改进节点法矩阵表示
Figure 1-5 The modified nodal representation of inductor L

图1-5是电感L在 $|sC(t) + G(t)|$ 矩阵中的表示。将带 s 的项放入 $C(t)$ 中，不含 s 的项放入 $G(t)$ 中，就可以利用QZ方法对其求解了。

电感还可以转化为一个回转器（gyrator）和一个电容的组合^[23]。这种组合会更有利于电路方程的标准化（normalization）。

用 QZ 方法对电路特征方程的求解过程在[22]中有详细的说明。当前，在[9]中的时变主极点根轨迹就是通过 QZ 方法得到的。

用 MATLAB 或者 LAPACK 数值求解包，可以将 C 和 G 两个矩阵分别转化为两个准上三角矩阵 T 和 S ，其中 $T = Q \cdot C \cdot Z$ ， $S = Q \cdot G \cdot Z$ 。其中 Q 和 Z 均为酉矩阵。特征方程的根可以由 T 和 S 的对角线元素求得：

$$\lambda_i = -\frac{S_{i,i}}{T_{i,i}} \quad (1-7)$$

其中， λ_i 为特征方程的第 i 个根。为了求解振荡器的周期时变主极点根轨迹，需要先对电路进行 PSS 分析并采样，然后线性化电路，得到 $C(t)$ 和 $G(t)$ 在一个周期中的采样值，对每个采样值进行求解，最后在所有电路极点中找出穿过虚轴的主极点，构成主极点根轨迹。

使用数值计算方法来计算振荡器主极点根轨迹的完整过程如下。

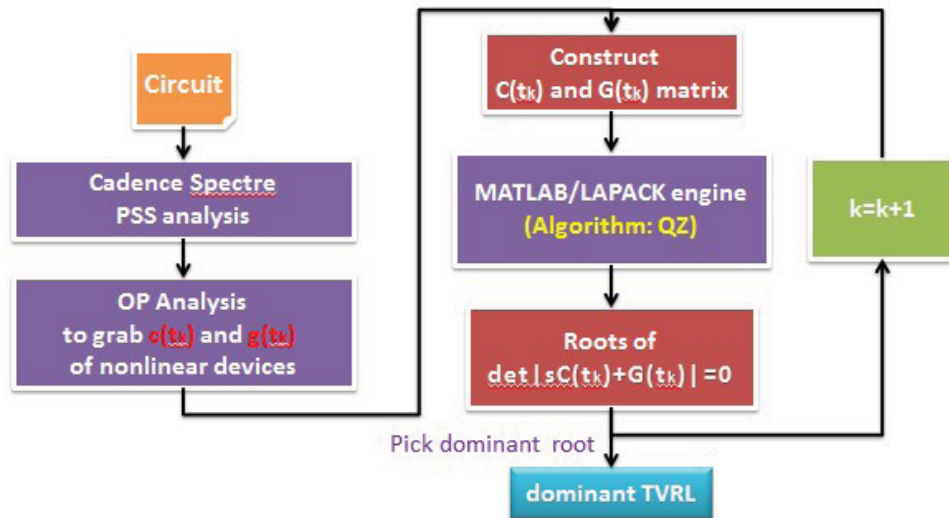


图 1-6 数值化振荡器主极点根轨迹计算过程示意图

Figure 1-6 Diagram of the flow to compute main time-varying root-locus of oscillators

具体来说，数值化的主极点根轨迹计算过程可以分为以下几步：

(1) 利用 Cadence Spectre 工具求解振荡器电路的周期性稳态(Periodic Steady State, PSS)解，并对其进行采样，得到电路中所有节点的电压。

(2) 代入节点电压，通过 Cadence Spectre 的直流工作点分析(DC Analysis)求解 MOS 管在 PSS 状态下每个采样点的小信号参数值。

(3) 在 PSS 状态下的每个时间点线性化整个振荡器电路，得到小信号网表。

(4) 求解每个时间点的小信号网表，得到式 (1-5) 中的 $C(t)$ 和 $G(t)$ 两个矩阵电路矩阵并通过 QZ 方法求解电路极点。从所求得的极点中，选择实数部分最大的共轭对，作为主极点。

(5) 不断迭代(4)中的过程，直到一个周期内的所有采样点处理完毕。

在这里，需要对主极点的选取做出一些解释。LC 振荡器有一个重要的特性，其输出是一个类似正弦的波形，比较纯粹，因此电路只可能受一对共轭主极点的影响，并且共轭主极点在电路振荡频率附近进行周期性的变化。其他极点均位于左半复平面，其中共轭的极点对均远离虚轴。因此，用(4)中的方法必定能选择出电路的主极点。

这种求根方法虽然速度快（复杂度为 $O(n^3)$ ），但是也有几个主要的缺点，具体表现在：

- (1) 随着时间的推移，电路中的小信号参数不断发生改变，因此 $C(t)$ 和 $G(t)$ 需要不断更新。在计算中，我们需要不断重新建立矩阵并求解，无法实现结构共享。
- (2) 矩阵求根方法无法揭示电路元件与系统方程及零极点之间的对应关系，所提供的知道信息有限，无法指出电路优化的方向。
- (3) QZ 方法求根会求解电路中所有的根，然而只有主极点是在电路分析中占重要地位的，其他极点不需求解。这里包含了一些计算冗余。
- (4) 相邻时间点的极点之间有密切的关系。然而 QZ 方法求解无法利用这一特点。我们可以找出一种方法，利用这一特点，对计算进行加速。

为了最大化地共享数据结构，对计算过程进行优化，并得到电路元件与系统方程之间的直接联系，我们很自然地联想到符号化的计算方法。

1.6 本文主要内容与章节安排

本文利用基于电路拓扑结构的符号化模拟电路分析算法 GPDD (Graph-Pair Decision Diagram) 以及 Muller 求根方法对高频振荡器的主极点进行求解，并在此基础上充分利用 GPDD 求解敏感度的便利性，得到高频电路振荡器主极点对于参数变化的敏感度，从而给出改善电路设计的关键信息。针对符号化仿真方法比较不同电路结构的不便性，本文提出了一种新的方法学，实现了电路元件的动态增加和减少，使得符号化仿真工具可以较为灵活地应对电路拓扑结构的更改。

本文一共分为六部分。

第一章阐述了电路分析的两类基本方法，以及对振荡电路进行周期性时变主

极点分析的数值方法。第二章介绍符号化图对决策树 GPDD (Graph-Pair Decision Diagram) 的基本概念以及用法。第三章详细介绍利用 GPDD 求解振荡电路时变主极点分析的符号化算法以及主极点敏感度算法。第四章介绍在 GPDD 结构的基础上, 对电路进行元件添加、删除的算法。第五章给出符号化求解振荡电路时变主极点以及添加、删除元件算法的实验结果及效率分析。第六章总结全文, 并提出展望。

第二章 符号化仿真方法简介

2.1 符号化仿真方法总体介绍

符号化计算方法已经比较长时间的历史。早在上世纪六、七十年代，随着计算机技术的发展，符号化的电路分析方法已经成为研究热点。符号化电路分析方法大概可以分为两个大类，分别是基于电路方程的代数方法及基于电路图结构的拓扑方法。具体细分，代数方法又包含了矩阵行列式法、参数提取法和数值插入法，而拓扑方法又包含了符号流图法、生成树枚举法等[12]。在这些方法中，矩阵行列式法及信号流图法被证明为比较有效和灵活的算法。但是，当时这些方法所能够解决的电路规模十分有限，只能解决大约15个节点，30条支路规模的电路。

2.2 二分决策图

近年来，符号化分析方法的发展使得电路分析的规模大大提升。而这些分析方法都是以二分决策图(Binary Decision Diagram, BDD)为基础的。

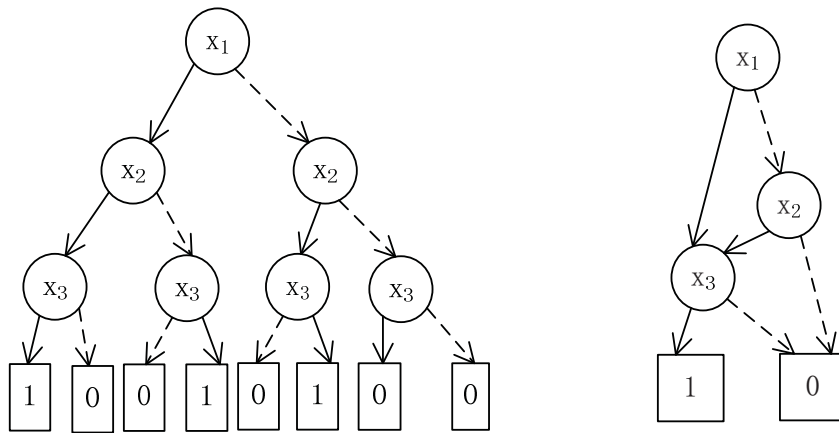
BDD 是一种有向无环图(Directed Acyclic Graph)，包含了两个终端节点及一系列非终端节点。两个终端节点为 0 和 1，非终端节点的出度为 2，分别为低端及高端，表示两种不同的决策。BDD 最早是为了解决布尔逻辑函数的求值及验证而诞生的。

1986 年，R. E. Bryant[13]提出了约化的二分决策图(Reduced Ordered Binary Decision Diagram, ROBDD)，并证明了同一个逻辑函数只有唯一的一个 ROBDD 表示，而在一定的符号排列顺序(order)下，这种表示是逻辑函数的最简形式，实现了所有相同子图的共享。1990 年，K.S. Brace 等人给出了一个 ROBDD 的有效实现[14]。ROBDD 使得 BDD 实现了内部结构的共享，一定程度上解决了 BDD 的复杂度随符号数量指数增长的问题，节约了大量内存空间。

我们用一个布尔运算来简要说明 ROBDD 的思想。真值表如图 2-1 所示。对应 BDD 如图 2-1a)所示。

F	x_1	x_2	x_3
0	0	0	0
0	0	0	1
0	0	1	0
1	0	1	1
0	1	0	0
1	1	0	1
0	1	1	0
1	1	1	1

图 2-1 示例真值表
Figure 2-1 An example true table



a) 示例布尔表达式的 BDD
a) BDD of the example Boolean function

b) 示例布尔表达式的 ROBDD
b) ROBDD of the example Boolean function

图 2-2 示例布尔表达式的 BDD 和 ROBDD
Figure 2-2 BDD and ROBDD of the example Boolean function

图 2-1 中，实线边表示对元素值取 1，虚线边表示对元素值取 0。综合图 2-1a) 中结果为 1 的路径，可以得到布尔表达式为 $x_1x_2x_3 + x_1\bar{x}_2x_3 + \bar{x}_1x_2x_3$ 。将图 2-1a) 中的相同结构进行最大化的共享，同时消除左右分支指向同一节点的冗余项，可以得到图 2-1b)，表达式为 $x_1x_3 + \bar{x}_1x_2x_3$ 。[13] 中详细说明了这一过程，并且证明了图 2-1b) 是在表达式相同，且符号顺序一定 ($x_1 \rightarrow x_2 \rightarrow x_3$) 的情况下唯一的规范化

(canonical) 最简形式。

ROBDD 有一个重要的特性, 即在符号顺序 (order) 不同的情况下, 数据结构的大小差别很大, 可能为多项式级别, 也可能为指数级别。一个例子可以说明这个问题。

假设现有表达式 $x_1x_4 + x_2x_5 + x_3x_6$ 。我们分别用两个顺序构造对应的 ROBDD。第一个顺序为自然顺序 $x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5 \rightarrow x_6$, 第二个顺序为 $x_1 \rightarrow x_4 \rightarrow x_2 \rightarrow x_5 \rightarrow x_3 \rightarrow x_6$ 。

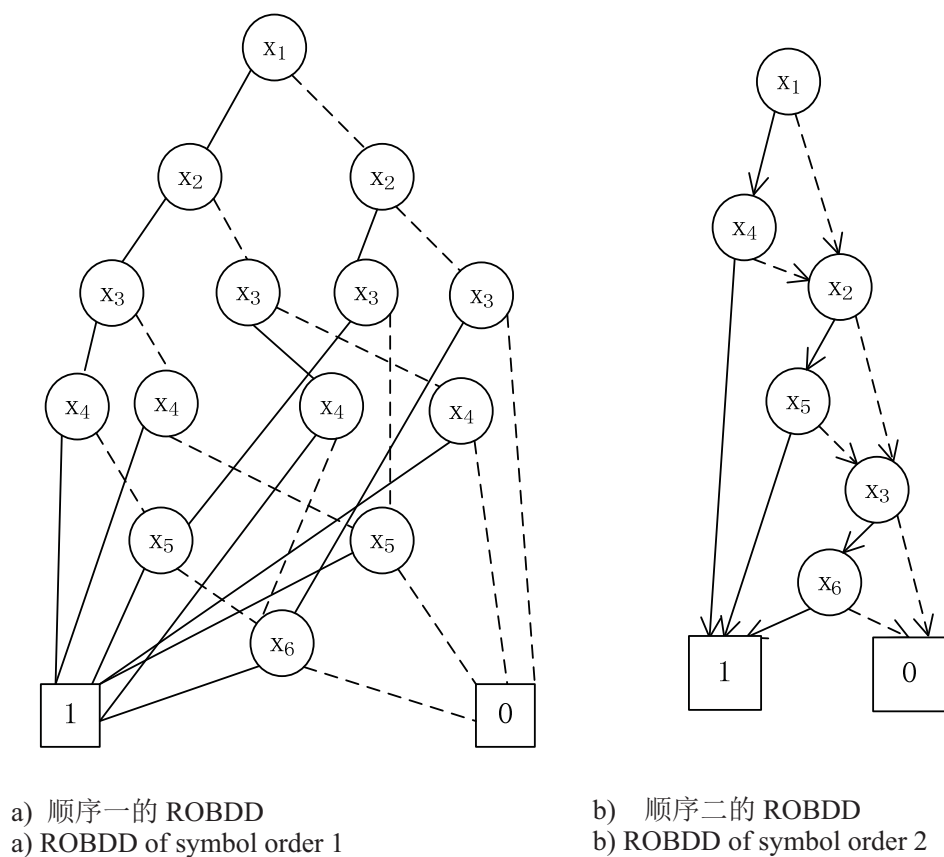


图 2-3 不同顺序的 ROBDD 大小比较
Figure 2-3 Comparison of ROBDD in different symbol order

从图 2-3 中, 我们可以看到不同符号顺序对 ROBDD 结构大小的巨大影响。决定最佳符号排序是一个 NP 问题^[24]。目前的符号化仿真器中, 符号顺序都是基于启发式算法决定的。

ROBDD 的思想不仅在布尔逻辑函数分析方面取得了重大成就, 同时也可应用在代数运算的分析上。在这一基础上, 进一步产生了许多符号化的模拟电路仿

真器。C. J. Shi 和 X. D. Tan[15]提出了一种基于行列式决策图(Determinant Decision Diagram, DDD)的符号化分析方法,运用 ROBDD 的数据结构,实现了电路解析公式中相同项的最大化共享,在一定程度上减慢了内存消耗,将符号化仿真器可分析的电路规模扩展 20 多个晶体管的数量级。

但是在这一仿真方法中,二分决策图中的节点和具体的电路元件并不是一一对应的。通常,一个元件会对应四个不同的节点。这将给灵敏度分析带来额外的开销。另外,这个结构所生成的乘积项会有相互抵消的情况。这会带来一些计算误差。如果检测这些抵消的乘积项,又需要引入另外的开销。

为了解决这一系列的问题,在严格的数学证明下,一套基于 GPDD(Graph Pair Decision Diagram) [16]的符号化分析方法应运而生,避免了行列式分析方法中可抵消项的产生。

一个符号化仿真器 GRASS (Graph Reduction Analog Symbolic Simulator) [17]就是基于这套理论实现的。这个仿真器仿真的电路规模也达到了 20 几个晶体管的级别。[18]给出了这个仿真器的设计细节。为了说明 GPDD 算法的基本原理,这里扼要地说明一些重要的规则,并给出示例。具体实现及证明,请参考文献 [16][17][18]。

2.3 GPDD 仿真算法基本原理

2.3.1 基本约束条件

GPDD 仿真工具可以处理四类基本的电路元件,分别是导纳(阻抗可通过取倒数转化为导纳)、受控源(压控电压源 VCVS、压控电流源 VCCS、流控电压源 C CVS、流控电流源 CCCS)、独立源(电压源、电流源)以及理想放大器。如果电路中含有非线性元件,则需要先转化为对应的小信号模型。对于受分析的电路,有以下几个约束条件。

- (1) 电路中只能包含一个独立源。超过一个独立源,则需要对每一个独立源进行单独分析,然后进行线性叠加。
- (2) 受控源的控制端以及受控端是一一对应的。每个控制端控制一个受控端,每个受控端受一个控制端控制。

对于受控源,也有几个约束条件。

- (1) 受控源的电压控制端(VC 边)电流为 0。
- (2) 受控源的电流控制端(CC 边)电压为 0。

2.3.2 有向图转换规则

在满足上述条件的基础上，我们就可以进行从原始电路到有向图的转换了。这一转换应该满足以下的规则。

(1) 将电路中的元件转化为有向边，从正极指向负极。电流源的有向边方向与电流方向相同。

(2) 压控源的电压控制支路增加一条并联的有向边（电流为 0），方向从电压的正极指向负极。

(3) 流控源的电流控制支路增加一条串联的有向边（电压为 0）。方向与电流方向相同。

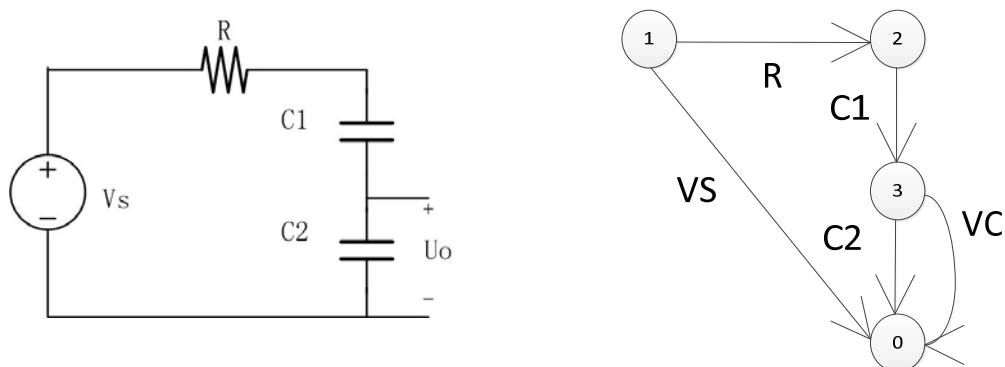
(4) 理想运算放大器用一个零器（Nullor）建模，每个零器由一个零阻器（Nullator, NU）和一个泛阻器（Norator, NO）组成。输入端用 NU 表示，输出端用 NO 表示。

(5) 将独立输入源抽象为输出端控制的受控源。用 X 表示。若输出为电压，则依据 (2) 并联添加表示输出电压的有向边。若输出为电流，则依据 (3) 串联添加表示输出电流的有向边。

规则 (5) 非常重要。为了计算电路的传输函数 $H(s)$ ，我们需要指定输入和输出端口。这一对应关系很容易让我们想到用受控源来表示模拟输入和输出的关系。为了保证电路的原始电气特性（每条支路的电流、电压）不受影响，我们将输出端指定为受控源的控制端，将独立源指定为受控源的受控端。电路传输函数可以表示为

$$H(s) = \frac{1}{X} \quad (2-1)$$

为了说明上述规则，我们用图 2-4 中的 RC 电路作为简单示例。



a) 一个简单的 RC 电路
a) A simple RC circuit

b) RC 电路的有向图
b) Directed graph created by the RC circuit

图 2-4 一个简单的 RC 电路及其有向图
Figure 2-4 A simple RC circuit and its corresponding directed graph

在 RC 电路对应的有向图中，输入输出受控源 X 包含 V_S 及 V_C 两条边。我们为输出端添加了并联的 V_C 边。由于 V_C 边电流为 0，电路特性不受影响。构建了有向图后，我们就可以开始构造 GPDD 结构了。

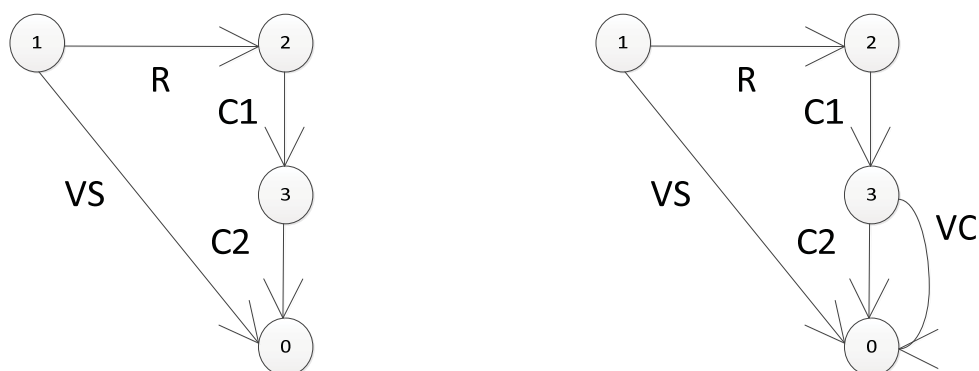
2.3.3 GPDD 构造规则

GPDD 是由一个图对开始构造的。根据我们的初始有向图，我们可以分别构建图对中的左图和右图。图中有些边只能出现在左图中，有些边只能出现在右图中。规则如下。

- (1) 左图无 NU 及 V_C 边。
- (2) 右图无 NO 及 CS 边。

因此，图 2-4 中所示结构的左图和右图应如图 2-5 所示。

我们需要在构建 GPDD 之前进行预处理操作，将初始的左、右图中包含的所有 NU 边及 NO 边短路。这是由 GPDD 的规则决定的。若电路包含 Nullor，则所有的 NU 边及 NO 边都必须包含在有效生成树对中。进行完这个预处理操作以后，我们就可以开始进行 GPDD 的构造了。



a) RC 电路左图

a) The left graph of the RC circuit

b) RC 电路右图

b) The right graph of the RC circuit

图 2-5 RC 电路有向图的左图和右图

Figure 2-5 Left graph and right graph created from the directed graph of the RC circuit

以图 2-5 中的两个子图作为根节点，我们可以开始构建 GPDD。GPDD 的构建过程是一个二分决策的过程，对电路中的元件按顺序进行操作，对子图进行约化。对每个元件的操作包含一条 1-路径和一条 0-路径。1-路径表示“包含”(Include)操作，0-路径表示“剔除”(Exclude)操作。具体的规则如表 2-1 所示。

表 2-1 GPDD 图约化规则

Table 2-1 Graph reduction rule of GPDD

	包含 (Include, 1-边)		排除 (Exclude, 0-边)	
	左子图	右子图	左子图	右子图
VCVS	短路 VS	断开 VS 短路 VC	短路 VS	短路 VS 断开 VC
CCVS	短路 VS 断开 CC	断开 VS 短路 CC	短路 VS 短路 CC	短路 VS 短路 CC
VCCS	短路 CS	短路 VC	断开 CS	断开 VC
CCCS	短路 CS 断开 CC	短路 CC	断开 CS 短路 CC	短路 CC
Y/Z	短路 Y/Z	短路 Y/Z	断开 Y/Z	断开 Y/Z

在图约化的过程中，有一点需要注意。我们在对边进行短路操作的规则如下。

- (1) 在图中留下序号较小的点。假设所短接的边连接 n_1 , n_2 两个点，且 $n_1 < n_2$ ，则留下 n_1 点。
 - (2) 对图中所有超过 n_2 的点，序号减一。假设有 $n_3 > n_2$ ，则 $n_3 := n_3 - 1$ 。
- 我们将第 (2) 步的操作定义为“保持连续性操作”。在[16]的规则中并未提

到这一操作。然而，这一操作是非常有必要的。在构造的过程中，我们需要对所生成的图进行哈希操作，以最大 GPDD 结构中的共享率，减少重复构造。判定两个图相同，需要两个图的节点序号完全一致，两个图的有向边完全一致，且两个图中对应有向边的入射点与出射点也一一对应。若缺少保持连续性操作，则会有同构子图被判定为不相同的情况出现。

我们用图 2-6 说明这一点。

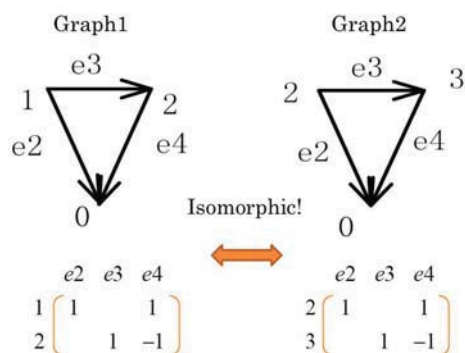


图 2-6 同构图检测

Figure 2-6 Isomorphic Graph Detection

如图 2-6 所示，Graph1 与 Graph2 中的边完全相同，但是点的具体编号有所不同。我们将图对应到相应的矩阵表达形式上，可以看出，两个矩阵的内容是一致的。GPDD 的构造过程是在矩阵中寻找有效生成树（对）的过程，规则与矩阵中的操作是完全对应的，因此，若对应矩阵相同，则我们应该判断两个图为相同的图。在 Graph2 对应的矩阵中，标号为 2 的行实际为第 1 行，标号为 3 的行实际为第 2 行。进行保持连续性操作，可以保证图中点的序号与其在矩阵中对应的实际行号一一对应。这样，Graph1 和 Graph2 就能够被检测为相同的图。

除了表 2-1 中所述规则以外，我们在进行包含 (include) 操作和排除 (exclude) 操作的同时，还需要确定对应操作的正负符号。由于我们进行了保持连续性操作，因此[16]中的符号决定规则可以进一步简化。我们将所要决定的符号记为 sign。

- (1) 初始化 sign=1; 如果当前是对 VCVS 或 CCVS 进行选取 (include)，则 sign*=-1。
- (2) 如果操作为断开某条边 (open)，则符号保持不变。
- (3) 如果操作为短路某条边 (short)，且假设这条边从 n2 指向 n1，则我们需要对 n1 和 n2 的大小进行判断。如果 n1>n2，则 sign*=-1，并将 n1 和 n2 置换，保证 n1<n2。若 n2 为奇数，则 sign*=-1。然后按前文所述的短路操作，保留 n1 节点，并保证剩余节点的连续性。

通过上述规则，我们就可以从图 2-5 中的左右图为起点，开始建立 GPDD 结构了。[18]中详细叙述了 GPDD 构造过程中的中止条件，使得构建结束于 0 端口和 1 端口，在这里不再赘述。

在建立 GPDD 结构时，为了实现结构的最大化共享，我们有三层的哈希机制。第一层是基于单个子图的哈希，用于检测同构的有向子图。第二层是基于有向子图对的哈希，在自顶向下的构造中使用，避免重复构造。第三层是基于 GPDD 节点三元组的哈希，自底向上，操作类似于从 BDD 到 ROBDD 的变化，检测 GPDD 中的重复结构并实现共享，使得 GPDD 变为规范化 (canonical) 的最简形式。这三层哈希机制在[18]均有详细说明。

根据上述构造过程，我们按照 $X \rightarrow C1 \rightarrow R \rightarrow C2$ 的符号顺序构建 GPDD 并进行哈希共享后，可以得到图 2-4 对应的 GPDD 结构，如图 2-7 所示。

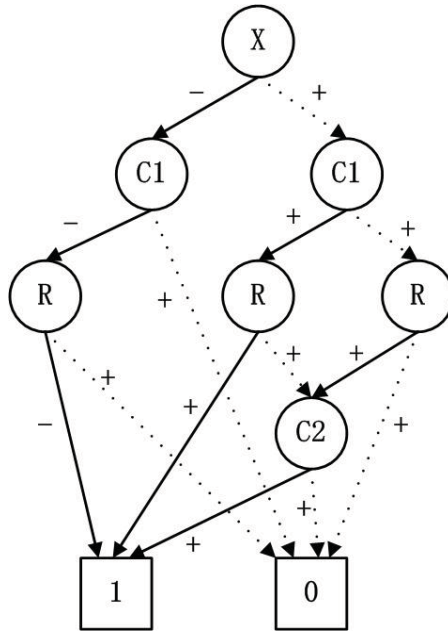


图 2-7 RC 电路对应的 GPDD
Figure 2-7 GPDD of the RC structure

在 GPDD 中，实线即为 1-路径，对应乘法运算，虚线即为 0-路径，对应加法运算。

对 GPDD 进行求值时，需要以深度优先遍历整个结构。求值过程如图 2-8 所示。我们用 $V(P)$ 表示以当前节点 P 为根节点的子 GPDD 的值， $S(P)$ 表示 P 的值 (若 P 为电阻、电感或电容，则需要转化为导纳值，可控源则不需要转化)，以 $V(C_1)$ 表示其左儿子 (1-路径指向的子 GPDD) 的值， $V(C_2)$ 表示其右儿子 (0-路径指向

的子 GPDD) 的值, 则有

$$V(P) = \text{Sign1} * V(C_1) * S(P) + \text{Sign0} * V(C_2) \quad (2-2)$$

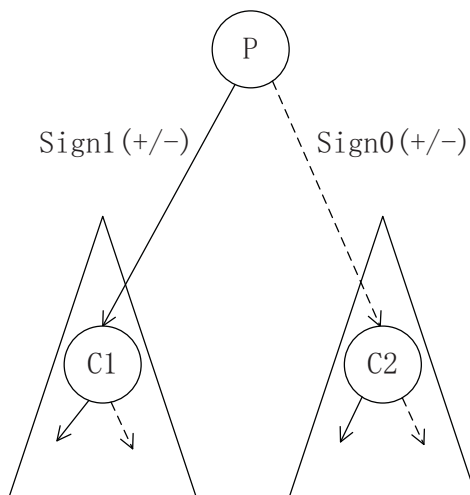


图 2-8 GPDD 求值过程

Figure 2-8 The evaluation procedure of GPDD

根据求值规则遍历 GPDD 结构, 可以在根节点处得到表达式 $-XC_1sR^{-1} + C_1sR^{-1} + C_1sC_2s + C_2sR^{-1}$ 。根据[16]中的理论, 这个表达式的值为 0。

$$-XC_1sR^{-1} + C_1sR^{-1} + C_1sC_2s + C_2sR^{-1} = 0 \quad (2-3)$$

这样, 我们就可以推出电路的系统方程:

$$H(s) = \frac{1}{X} = \frac{C_1R^{-1}s}{(C_1R^{-1} + C_2R^{-1})s + C_1C_2s^2} \quad (2-4)$$

可以看出, $H(s)$ 的分母对应 GPDD 根节点 0-路径下的结构, 而分子对应根节点 1-路径下的结构。

$H(s)$ 的分母 $(C_1R^{-1} + C_2R^{-1})s + C_1C_2s^2$ 即为电路的特征多项式。实际上, 可以通过 GPDD 的规则证明电路的特征多项式与输入输出无关。

为了说明这一问题, 我们依然以图 2-4 中的 RC 电路作为例子。为了忽略电压源以及输出端口的影响, 我们将电压源置 0 短接, 不指定输出。这样, 可以得到如图 2-9 所示的电路。

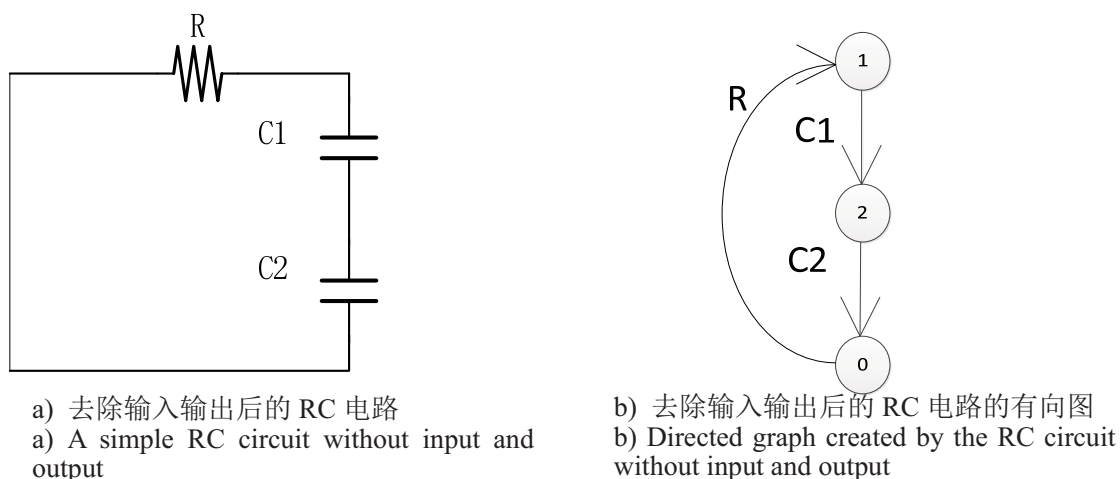


图 2-9 去除输入输出后的 RC 电路及其有向图

Figure 2-9 A simple RC circuit and the corresponding directed graph without input and output

按照规则，由图 2-9b) 中的有向图产生的左右两个子图均与其一致。

再对图 2-5 中的 VCVS 进行排除 (exclude) 操作，可以得到图 2-7 中 GPDD 根节点 0-路径指向的子 GPDD 的两个子图对。我们发现，这两个子图对与图 2-9b) 中的结构亦是完全一致的。

这就说明了电路的特征多项式与输入输出的位置无关。

因此，在不添加输入输出的情况下，对电路进行 GPDD 构造，可以直接得到电路的特征多项式。

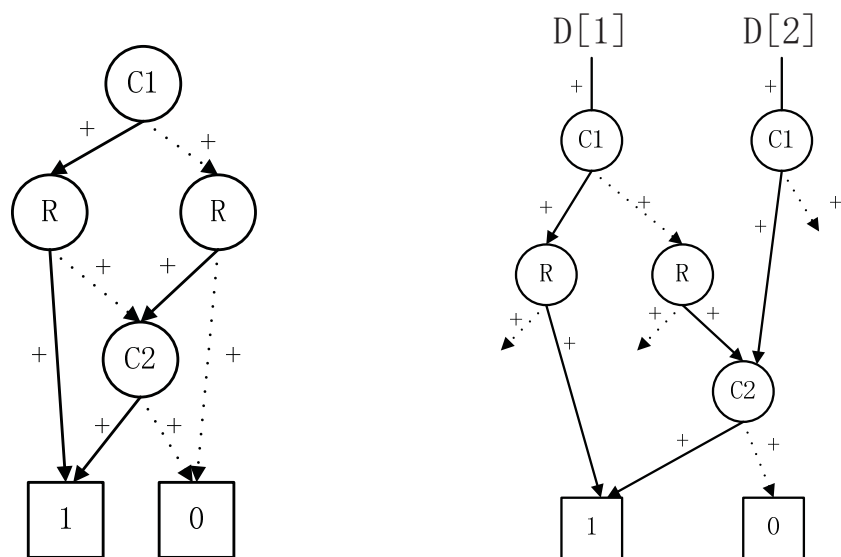
2.3.4 GPDD 的 s 展开

GPDD 是一个复数结构。如果用 GPDD 对电路进行 AC 响应的分析，需要重复地将不同频率点的采样值代入结构中电容与电感，对结构进行多次遍历。如果频率采样点很多，这一计算效率就比较低了。因此，我们希望通过一种数据结构，得到 s 的系数表达式。这样，在计算频响时，我们可以分两个步骤进行计算。

(1) 遍历数据结构，求解 s 的多项式系数。

(2) 将频率值代入多项式中，得到频响。

实际上，这种数据结构可以很方便地从 GPDD 产生。[18]给出了由 GPDD 结构生成一种 s 展开 BDD 结构的方法，用以储存系统方程的分子、分母多项式系数。在这里，我们给出 RC 电路特征多项式的 GPDD 结构以及它所对应的 s 展开 BDD 结构，如 2-10 所示。



a) RC 电路特征多项式的 GPDD 结构
a) GPDD of RC circuit's characteristic polynomial

b) RC 特征多项式的 s 展开 BDD 结构
b) s-expanded BDD of RC circuit's characteristic polynomial

图 2-10 RC 电路特征多项式的 GPDD 结构及 s 展开 BDD 结构

Figure 2-10 The GPDD and s-expanded BDD structure of RC circuit's characteristic polynomial

图 2-10b) 结构中未画出末端的虚线均指向 0 终端。D[1] 及 D[2] 分别表示特征多项式中 s 的 1、2 阶项系数。其求值过程与图 2-8 中类似。不同的是，在这个结构中，电容无需化为频域形式，电阻和电感元件均取元件值的倒数。对这个结构进行深度优先遍历求值，可以得到 1 阶系数 $C_1 R^{-1} + C_2 R^{-1}$ 与 2 阶系数 $C_1 C_2$ 。得到这些系数以后，我们就可以求解特征多项式的根了。

2.4 本章小结

本章先对 BDD 数据结构以及其对应的最简规范化形式 ROBDD 进行了简单介绍，然后介绍了基于 BDD 的符号化仿真工具 GPDD。GPDD 的构建基于对图的操作，其实现原理较为复杂，本章给出了简要的介绍，并在原本的构建规则上给出了一些优化改进方法。不同于数值化分析方法，GPDD 是基于电路的拓扑结构进行构造的。在电路拓扑结构固定的情况下，利用 GPDD 进行求解分析可以带来一些优势，包括更少的循环迭代求解时间以及更加便利的敏感度求取等。GPDD 的这些优势，对于振荡器根轨迹的求取，是很有意义的。

第三章 符号化振荡电路时变主极点根轨迹分析方法

3.1 总体流程

回顾第一章中的周期性时变线性系统模型，用数值方法对振荡电路进行时变主极点根轨迹分析时，需要将其转化为线性时变电路，进而求解其周期性时变特征多项式 $D(s,t)$ 的根，从中挑选主极点。

这一过程主要可以分为一个大信号部分和一个小信号部分。首先，我们对电路进行大信号周期性稳态分析，并将其线性化为时变电路。其次，我们对线性化后的电路模型提取特征多项式对应的两个矩阵，并通过数值方法求根。

用符号化分析方法计算振荡电路的大信号过程与这一过程类似，而在小信号部分，与数值方法建立矩阵进行循环 QZ 分析的方法不同，我们为小信号电路建立 GPDD 结构，并利用这一结构的优点，对电路进行迭代求根。

总体分析流程如图 3-1 所示。

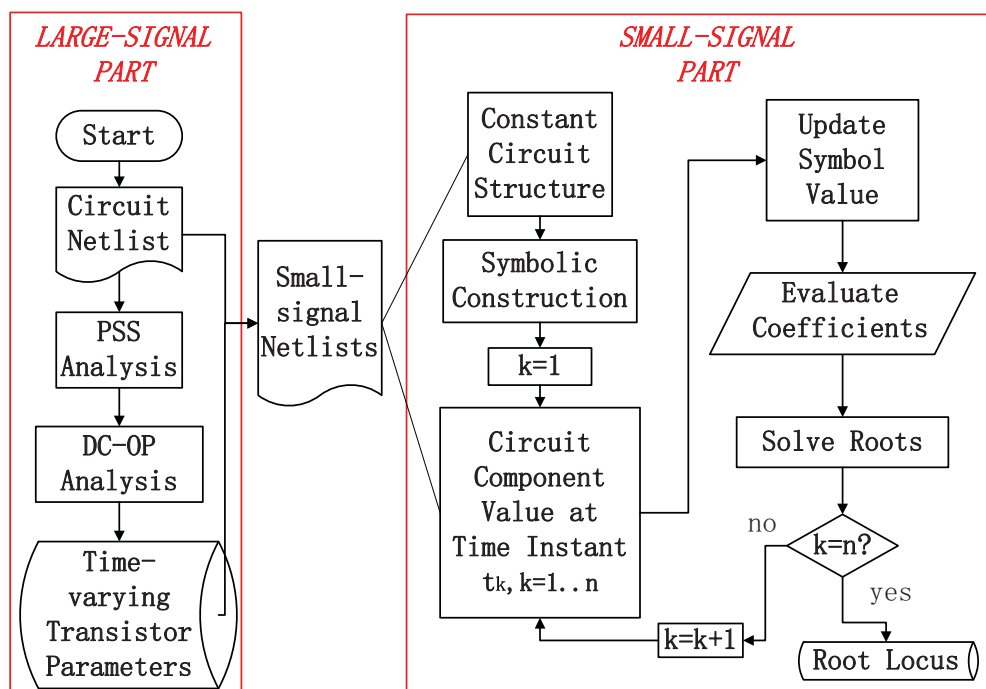


图 3-1 符号化振荡电路主极点根轨迹分析流程

Figure 3-1 The process to calculate symbolic time-varying root-locus of oscillators

下面我们将详细说明这一流程。

3.2 大信号部分分析流程

3.2.1 PSS 仿真条件

要求得振荡器的根轨迹，首先我们需要对其进行大信号周期性稳态解 PSS 求解，以求得振荡器的时变工作点。当前能够求解振荡器周期性稳态解的商用仿真器有很多，包括 HspicRF^[25]、Cadence Virtuoso Spectre^[26]等等。当前比较流行的仿真算法和工具可以在^[27]中找到。在本文中，我们使用 Cadence Virtuoso Spectre 求解振荡器的 PSS 解。

用 Spectre 求解 PSS 时，仿真工具能够自动保存在一个振荡周期 T 内，在所有采样时间点 t_k 上，电路中的节点电压向量 $V(t_k)$ 。 $V(t_k)$ 为一个多维向量，其维度与电路中的节点数相等。如果在一个 PSS 周期内共有 n 个采样时间点，则这样的 $V(t_k)$ 亦有 n 个。我们需要将这一数据保存下来，并用这一数据将电路线性化。

3.2.2 线性化电路

在我们的振荡器中，包含的非线性元件只有 MOSFET 一种。求得时间节点 t_k 上的电压向量 $V(t_k)$ 后，我们需要将这一向量加到对应的 MOSFET 端口，对其进行工作点 DC-OP 分析，如图 3-2 所示。这一步骤可以通过编写 Cadence 提供的 SKILL 语言代码自动完成。

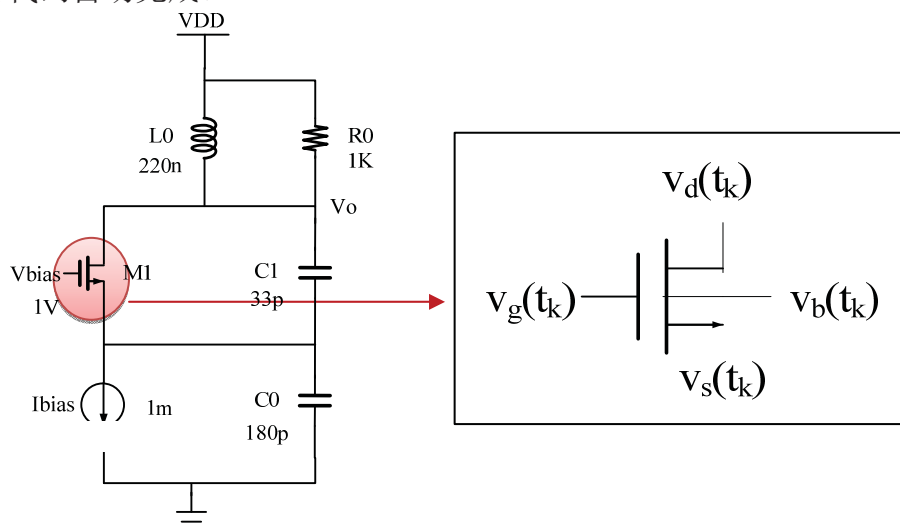


图 3-2 DC-OP 工作点分析示意图

Figure 3-2 Sketch map of DC-OP analysis

DC-OP 分析将提供在时间节点 t_k 上，电路中所有 MOSFET 的小信号变量值。我们使用的小信号模型为^[28]中的 MOS 完整准静态模型，如图 3-3 中所示。这一

模型能够较为精确地仿真高频电路 MOS 管至 1/3 特征频率处。同时，这一模型与 Spectre 仿真器 DC-OP 分析提供的小信号参数相符合。

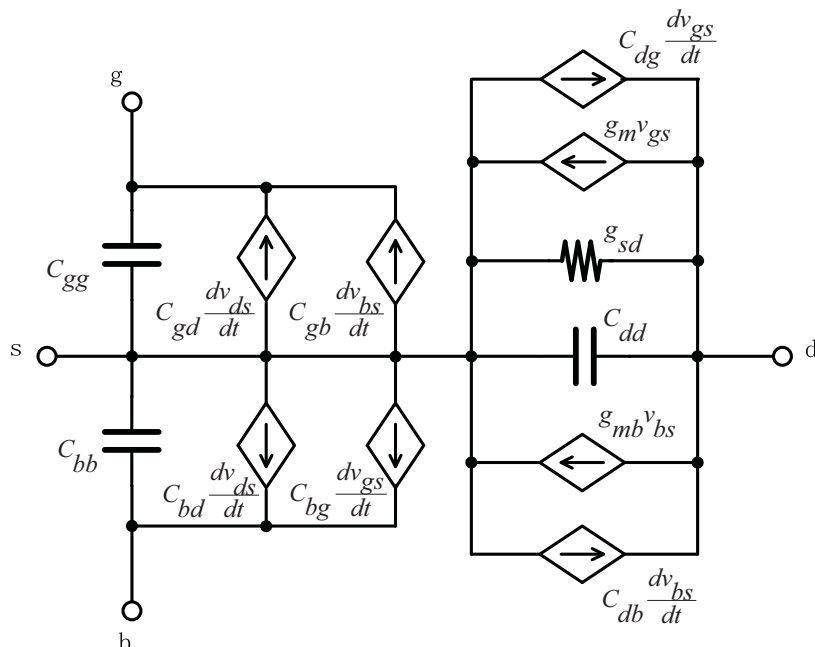


图 3-3 MOS 管完整准静态模型^[28]
Figure 3-3 Complete quasi-static model of MOSFET

我们注意到，准静态模型中出现了六个特殊的元件 $C_{gd} \frac{dv_{gs}}{dt}$ ， $C_{gb} \frac{dv_{bs}}{dt}$ ， $C_{bd} \frac{dv_{ds}}{dt}$ ， $C_{bg} \frac{dv_{gs}}{dt}$ ， $C_{dg} \frac{dv_{gs}}{dt}$ ， $C_{db} \frac{dv_{bs}}{dt}$ 。这些元件我们称之为电容跨导。电容跨导与跨导（VCCS）类似，只是其跨导值为 C_s ，而非我们通常所见的 G 。用数值方法处理时，我们需要将这六个电容跨导映射到矩阵 $C(t)$ 中。用符号化工具 GPDD 求解时，对这种元件的处理，有两点需要注意。

- (1) GPDD 构造时，将电容跨导视为 VCCS，适用表 2-1 中 VCCS 的规则。
- (2) 对 GPDD 进行 s 展开时，用电容跨导视为电容 C_s 。

在时间点 t_k 上，将振荡器电路中的 MOS 管替换为图 3-2 中的小信号模型，代入 DC-OP 得到的小信号参数值，同时将电压源短路、电流源开路，我们就完成了对振荡器电路的线性化。

3.3 小信号部分分析流程

3.3.1 数据结构的建立

在上述将振荡器从大信号电路转化为小信号电路的过程，我们注意到，在一个振荡周期内，振荡器电路的结构始终保持不变。因此，它所对应的小信号电路也保持恒定的拓扑结构。

若用数值化方法分析，则我们无法利用这一特点。数值化分析方法基于电路中的小信号参数值，在一个振荡周期内，随着时间的推移，我们得到 PSS 不同时间点的采样值， $V(t_k)$ 不断更新，因此 MOS 管的小信号参数值也不断更新。因此，在求解新的时间点时，我们需要重新构造时变特征多项式对应的两个矩阵。

然而，符号化的数据结构则可以很好地利用电路拓扑结构不变这一特点。由于符号化的数据结构是基于电路拓扑结构而构建的，与电路元件参数值无关，因此，我们在求解时变根轨迹时，只需要根据小信号电路对 GPDD 进行一次构造，并对其进行 s 展开，得到 s 展开 BDD 结构，则可重复利用它进行时变特征多项式的分析。这会给计算带来很多便利。

假设 PSS 分析结果有 n 个时间采样点，则符号化小信号分析的步骤如下。

- (1) 根据小信号电路拓扑结构构造 GPDD，并以此为基础构建 s 展开 BDD 结构。
- (2) $k=1$ 。
- (3) 代入时间点 t_k 的小信号参数值，对结构进行深度优先遍历求解，得到特征多项式 $D(s, t_k)$ 的系数。
- (4) 得到系数后，对 $D(s, t_k)$ 求根，提取主极点。
- (5) $k=k+1$, $k>n$ ，结束，否则返回步骤 (3)。

3.3.2 周期时变特征多项式系数求解

我们仍然以一个无源的 RC 时变电路为例来说明这一过程。如图 3-4 所示，我们假设 R1, C1 和 C2 均为非线性元件的小信号参数值。每拿到一个新的采样点，电路中元件 R1, C1 和 C2 的值都需要更新。GPDD 结构有一个符号列表，用以存放结构中所有元件的值。我们需要对列表中的值进行刷新。电路中，线性元件的值是非时变的，而只有非线性元件的小信号参数值是时变的，若电路中同时包含这两种元素，则我们只需要更新非线性元件的值。

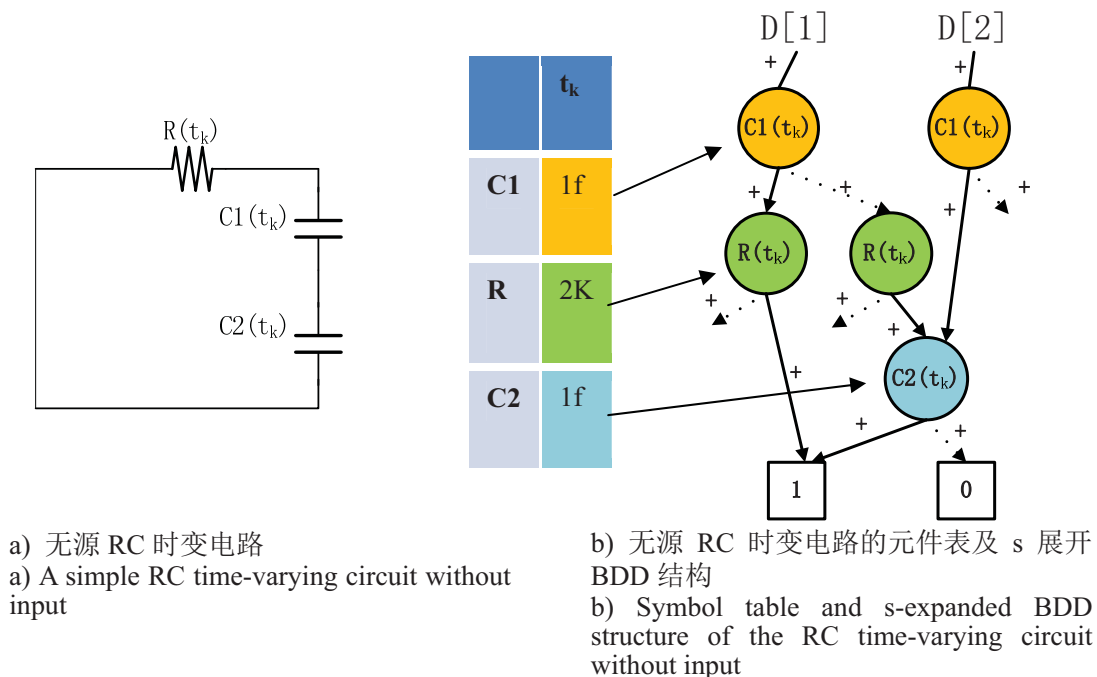


图 3-4 无源 RC 时变电路及对应的 s 展开 BDD 结构
Figure 3-4 A simple passive RC time-varying circuit and its corresponding s-expanded BDD

更新了元件值以后，就可以对 s 展开 BDD 结构进行深度优先的求值运算了。这样，我们得到了特征多项式 $D(s, t_k)$ 的 s 系数值。

3.3.3 连续 Muller 主极点求解方法

得到了多项式的各项系数值，就可以通过这些系数值求根。某个时刻的系统特征多项式 $D(s, t_k)$ 可以表示为一个 N 阶多项式。特征多项式的阶数为电路中的电容总数加电感总数（在没有并联的情况下）。

给定一个 n 阶多项式，由代数基本定理，我们可以知道，多项式在复数域内有 n 个根（包含重根）。对于高于三阶的多项式，求方程的根通常运用迭代逼近的方法，产生一组根的序列，使得根的序列不断收敛于方程的根。在这些迭代方法中，以 Muller 方法[19]最为常用，它具有收敛速度快，能够求解复根的特点。

Muller 求根方法是一种利用抛物线进行牛顿插值，迭代求根的方法。它以 3 个初始值为起点，进行循环迭代。这三个初始值可以任意选取。基本原理如图 3-5 所示。

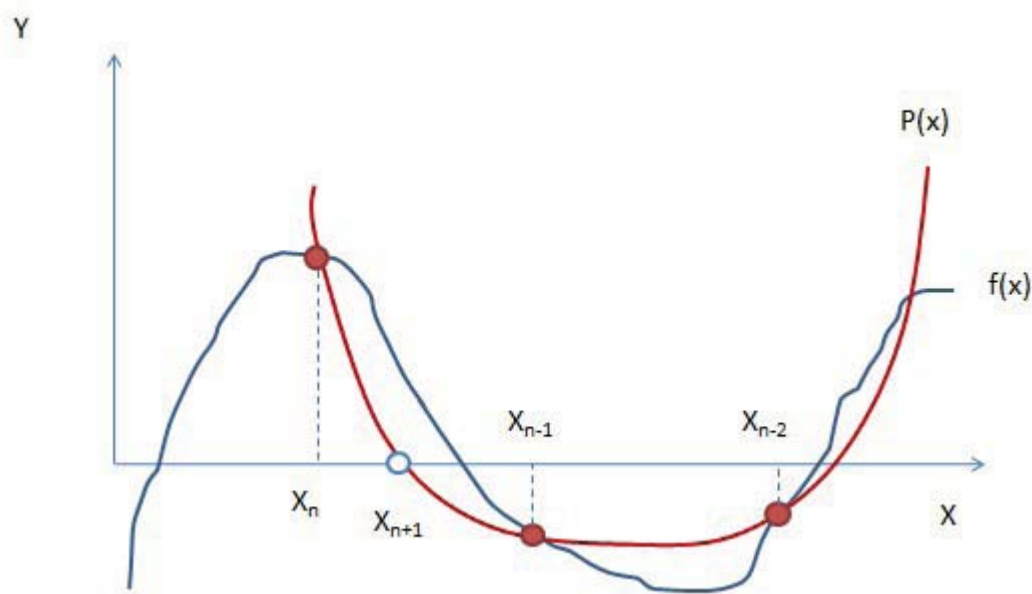


图 3-5 Muller 迭代过程示意图
Figure 3-5 Sketch map of Muller iteration

假设我们的当前迭代点为 $(x_{n-2}, f(x_{n-2}))$, $(x_{n-1}, f(x_{n-1}))$ 以及 $(x_n, f(x_n))$ 。我们可以作过这三个点的抛物线，并用牛顿插值法来表示它。假设这条抛物线为多项式 $P(x)$ ，则我们可以得到 $P(x)=0$ 的两个根（可能为复数）。选取两个根中离 x_n 最近的根，记为 x_{n+1} ，则我们可以通过点 $(x_{n-1}, f(x_{n-1}))$ ，点 $(x_n, f(x_n))$ 以及点 $(x_{n+1}, f(x_{n+1}))$ 进行下一轮的迭代，直到迭代点收敛于 $f(x)=0$ 的根。

这里不给出详细推导过程，用简单的几个步骤。说明 Muller 迭代的计算方法。我们定义

$$q = \frac{x_n - x_{n-1}}{x_{n-1} - x_{n-2}} \quad (3-1)$$

且有

$$\begin{cases} a = qy_n - q(q+1)y_{n-1} + q^2y_{n-2} \\ b = (2q+1)y_n - (q+1)^2y_{n-1} + q^2y_{n-2} \\ c = (q+1)y_n \end{cases} \quad (3-2)$$

则可以推导出

$$x_{n+1} = x_n - (x_n - x_{n-1}) \frac{2c}{b \pm \sqrt{b^2 - 4ac}} \quad (3-3)$$

在式 (3-3) 中，正负号的选取取决于哪个根离 x_n 更近。我们选择使

$|b \pm \sqrt{b^2 - 4ac}|$ 更大的值。

在迭代求解的过程中，我们需要注意，式 (3-1) 中的 q 不能为 -1，否则我们的计算公式会出现除 0 的情况。亦即，我们需要保证 x_n 与 x_{n-2} 不相等。若出现相等的情况，我们可以给迭代数据添加一些偏移，以保证求值的顺利进行。

Muller 迭代求解的收敛阶数为 1.84。

在求得了一个根以后，我们需要对多项式进行降阶。如果求得的根为实数，多项式降一阶。若求得的根为复数，则由于电路的特征多项式是实系数多项式，这个复根的共轭值也必然为多项式的根，此时多项式降两阶。

由于求得的根精度是有限的，因此，在对多项式进行降阶的过程中，计算精度将会越来越低，偏差会越来越大。为了尽量减少精度的损失，在降阶时，我们需要考虑使用前向降阶或者后向降阶方法。前向降阶方法是指求解降阶后的多项式时，从高阶项向低阶项求值。后向降阶方法是指求解降阶后的多项式时，从低阶项向高阶项求值。这里对详细求解过程不再赘述。具体可参见[19]。

在求得的根数值较小时，我们通过前向降阶方法进行降阶。根数值较大时，要通过后向降阶方法进行降阶。在实现中，我们设定两个方法的根分界线为 10^{15} 。

降阶完成后，我们可以继续对多项式进行迭代求解，求得下一个根。

虽然 Muller 方法可以求得很高的精度，但是由于降阶会带来多项式系数本身精度上的损失，因此用 Muller 方法求得的根亦会有精度上的损失。因此，在求得每一个根时，我们将根代入未降阶的原始多项式求值，若小于阈值（定义为 10^{-4} ），则使用 2 阶收敛的牛顿迭代法，在未降阶的原始多项式基础上对根进行进一步的抛光，提升根的精度。

当多项式的阶数小于或等于 3 时，我们就可以根据公式对剩下的根进行直接求解，无需继续降阶。

在我们的应用中，只有振荡器的主极点根轨迹需要关心。并且，正弦 LC 振荡器符合一个条件，穿越虚轴的共轭根轨迹有且只有一对，且电路中的其他共轭复根均位于原理虚轴的左半平面。我们可以利用这一特点，进一步对我们的求解进行加速。

我们求解的是根轨迹。实际上，PSS 求解过程的采样点时间间隔比较小，因此，对于两个相邻的时间点，线性化电路时所使用的电压值很接近，导致 MOS 管小信号参数值也非常接近。对 GPDD 进行求值时，由于元件参数相对变化值不大，因此得到的 s 展开系数的相对变化值也较小。这就导致了相邻两个时间采样

点的系统特征多项式 $D(s, t_k)$ 与 $D(s, t_{k+1})$ 的系数相对非常接近（考虑到系数的阶数）。进而，根据这些系数所计算的主极点在空间上的分布也是非常相近的。考虑到正弦 LC 振荡器的特殊性，我们知道，特征多项式的其他根均分布在远离主极点的位置，因此，我们可以利用上一时间点的迭代结果，快速计算下一时间点的主极点。

具体地，我们进行如下的计算步骤。假设时间点总数为 n 。

- (1) 对第一个时间点 t_1 ，我们求得 $D(s, t_1)$ 的所有根，并从中挑选出实部最大，且虚部为正的复数根，即主极点。我们保存求解这一复数根时的最后三个迭代值。 $k = k + 1$ 。
- (2) 对 t_1 以后的所有时间点 $t_k, k > 1$ ，以前一个时间点 $D(s, t_{k-1})$ 的最后 3 个 Muller 迭代值为初始值，迭代求解 $D(s, t_k)$ 的主极点，同时保存当前求解过程的最后三个迭代值。若 $k = n$ ，则停计算完毕。否则， $k = k + 1$ ，重复步骤 (2)。

我们将这一过程称为连续 Muller 主极点求解过程。

图 3-6 详细说明了这一过程。

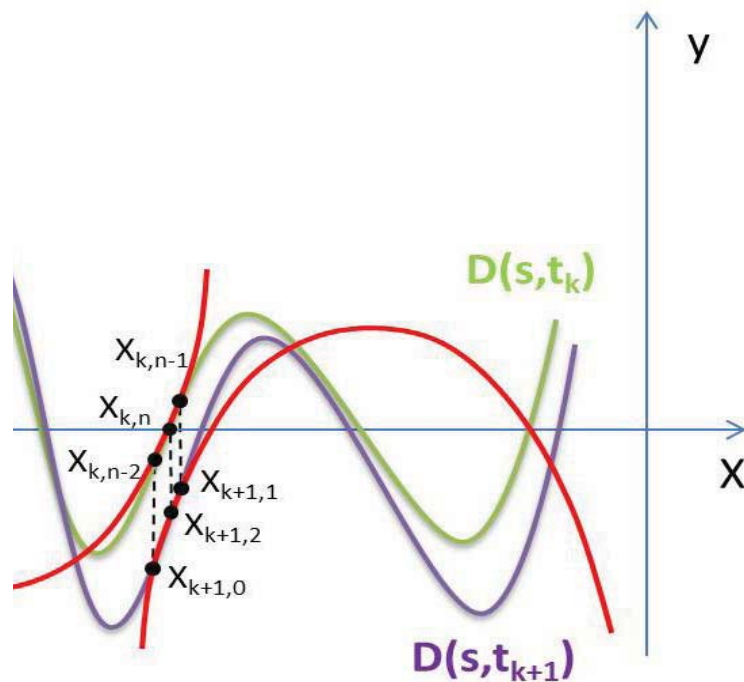


图 3-6 连续 Muller 主极点求解过程示意图

Figure 3-6 Sketch map of Successive Muller main root calculation process

如图所示,对 $D(s,t_k)$ 进行 Muller 迭代,收敛前的最后三个值为 $x_{k,n-2}$, $x_{k,n-1}$ 以及 $x_{k,n}$, 其中 $x_{k,n}$ 为 $D(s,t_k)=0$ 的主极点。令 $x_{k,n-2} = x_{k,0}$, $x_{k,n-1} = x_{k+1,1}$, $x_{k,n} = x_{k+1,2}$, 以这三个值为初始值进行 Muller 迭代求解,则我们可以在 4-5 个循环内得到 $D(s,t_{k+1})=0$ 的主极点。这一求解过程非常迅速。这是由于, Muller 方法的效率极大地取决于初始值的选取,由于两个相邻时间点的主极点值很接近,因此初始值的选取非常便利。

总结上述过程,在第一个时间点,我们需要不断进行迭代求根与降阶的操作,并从中挑选出我们需要的主极点。从第二个时间点开始,我们只需要进行迭代求根的运算即可,并且可保证迭代过程收敛于新的主极点。

这一效率的提高是由主极点根轨迹的特殊性以及 Muller 迭代方法的特性决定的。若利用数值 QZ 方法,则需要在整个求解周期内,求解所有时间点上的所有极点值,并且从中挑选出每个时间点上的主极点,练成主极点根轨迹。这一过程需要计算我们不需要的冗余信息,因此效率不及我们基于多项式的方法高。

综上所述,在利用 GPDD 对振荡器主极点进行分析时,我们首先需要根据电路的大信号分析,得到振荡器在每个时刻点上的小信号参数值,并利用一个恒定不变的电路拓扑及 s 展开 BDD 结构,求得每个时刻点上系统特征多项式的各项系数值。进而利用这一系数值以及连续 Muller 迭代方法进行主极点的求解。在对所有时刻点处理完毕后,我们就能够得到振荡器的主极点时变根轨迹了。

3.4 根轨迹敏感度分析方法

利用 GPDD 结构的特殊性,我们可以很方便地对系统函数进行敏感度分析,从而指导模拟电路的设计。[29]提出了用 GPDD 提取敏感度信息的算法及在模拟电路分析上的指导作用。[30]考虑了 MOS 管宽度变化对工作点的影响,所提取的敏感度信息比[29]更为精准,能够为模拟电路设计提供优化参考。

在这里,我们希望通过敏感度信息,查看在电路元件值发生变化的情况下,振荡器主极点时变根轨迹的变化趋势。我们用 s 表示极点的值,用 p 表示一个元件参数。则我们定义主极点对元件参数的敏感度如下。

$$Sens(s, p) = \frac{\partial s}{\partial \ln p} = \frac{\partial s}{\partial p} = p \frac{\partial s}{\partial p} \quad (3-4)$$

这一定义与[29]中的定义有所区别。其中元件 p 的数值作了归一化处理,提供相对变化的数值。对于 s 我们则不作此处理,这是由于我们希望在仿真结果中,

看到极点 s 在幅度和相位上，相对于元件 p 的绝对变化情况。

s 和 p 之间没有直接的对应关系。我们需要通过特征多项式为中介，求解这一敏感度信息。

在时刻 t_k ，我们有特征多项式

$$D(s, t_k) = a_0(t_k) + a_1(t_k)s + a_2(t_k)s^2 + \cdots + a_n(t_k)s^n = 0 \quad (3-5)$$

对 p 求导，我们有

$$\frac{\partial a_0(t_k)}{\partial p} + \frac{\partial a_1(t_k)}{\partial p}s + a_1(t_k)\frac{\partial s}{\partial p} + \frac{\partial a_2(t_k)}{\partial p}s^2 + 2sa_2(t_k)\frac{\partial s}{\partial p} + \cdots + \frac{\partial a_n(t_k)}{\partial p}s^n + na_n(t_k)s^{n-1}\frac{\partial s}{\partial p} = 0 \quad (3-6)$$

提取项 $\frac{\partial s}{\partial p}$ ，我们有

$$\frac{\partial s}{\partial p} = \frac{\frac{\partial a_0(t_k)}{\partial p} + \frac{\partial a_1(t_k)}{\partial p}s + \frac{\partial a_2(t_k)}{\partial p}s^2 + \cdots + \frac{\partial a_n(t_k)}{\partial p}s^n}{a_1(t_k) + 2sa_2(t_k) + \cdots + na_n(t_k)s^{n-1}} \quad (3-7)$$

我们发现，式 (3-7) 包含两部分的求解。第一部分是等式右边的分子，需要求得 $D(s, t_k)$ 所有系数对参数 p 的导数值，并代入当前求得的极点值 s ，从而求得分子数值。

第二部分是分母，可以直接利用 $D(s, t_k)$ 的系数值及当前极点值 s 进行求解。

实际上，每一个系数 $a_m(t_k)$ 可以表示为一个关于 p 的代数表达式

$$a_m(t_k) = p \bullet (A) + B \quad (3-7)$$

其中， A 和 B 表示含有加法和乘法运算的代数表达式。 $a_m(t_k)$ 是一系列乘积项之和，其中一些乘积项含有 p ，另一些乘积项与 p 无关。由 GPDD 的构建规则，我们知道一个元件 p 在乘积项中只可能出现一阶或零阶。我们将一阶项提取出来，即为 $p \bullet (A)$ ，剩下的零阶项即为 B 。

因此，对 p 求偏导数，我们可以得到

$$\frac{\partial a_m(t_k)}{\partial p} = A \quad (3-8)$$

因此，求解这项导数时，我们需要遍历 s 展开 BDD 结构，寻找包含 (include) p 元件的路径，忽略不包含 p 的路径。并在包含 p 的路径中，将 p 的值置为 1。

需要注意， s 展开 BDD 是有一个固定的符号顺序的，位于元件 p 以上位置元件，符号顺序在 p 之前。位于元件 p 以下的元件，符号顺序在 p 之后。我们从 s 展开 BDD 的根部开始进行深度优先遍历，定义当前遍历的节点 C ，左儿子为 C_1 ，

右儿子为 C_2 ， $Sign1$ 为 C 指向 C_1 的符号， $Sign2$ 为 C 指向 C_2 的符号。 $S(C)$ 为当前节点的符号在 s 展开 BDD 中等效的计算数值。那么，有三种情况。

- (1) C 的序号小于 p 的序号。若 C_1 序号大于 p 的序号，则 $V(C_1)=0$ ，否则，求解的值。若 C_2 序号大于 p 的序号，则 $V(C_2)=0$ ，否则，求解 $V(C_2)$ 的值。 $V(C) = Sign1 * V(C_1) * S(P) + Sign0 * V(C_2)$ 。
- (2) C 的序号等于 p 的序号。求解 $V(C_1)$ 的值。 $V(C) = Sign1 * V(C_1)$ 。
- (3) C 的序号大于 p 的序号。迭代求解 $V(C_1)$ 的值以及迭代求解 $V(C_2)$ 的值。 $V(C) = Sign1 * V(C_1) * S(P) + Sign0 * V(C_2)$ 。

这里有几点需要注意。

- (1) 对终端 1 以及终端 0，我们假设它们的序号大于 s 展开 BDD 中的所有符号。
- (2) 并联的元件共享相同的序号值。
- (3) 对已经求过值 V 的节点，我们给它们设立一个标记，避免重复求解。
若未求解过值 V ，则需要对当前节点做深度优先遍历求解，规则与上述提到的三种情况规则一样。

我们根据上述的规则，对整个 s 展开 BDD 从根部开始进行迭代求解，则可求得式 (3-8) 中的 $\frac{\partial a_m(t_k)}{\partial p}$ ，其中 $m = 0, 1, 2, \dots, n$ ， n 为 $D(s, t_k)$ 的阶数。若 s 展开 BDD

不包含某一阶的系数，则这一系数为 0。

将系数的导数代入 (3-7) 式，并将 (3-7) 的结果代入 (3-4)，我们就可以得到 $D(s, t_k)$ 的主极点相对于元件 p 的敏感度。扫描一个周期内的所有 $D(s, t_k)$ ，则我们可以看到根轨迹相对于元件 p 的变化趋势。

3.5 本章小结

本章对符号化振荡器时变主极点分析的整个流程作了详细的叙述。首先，利用 Cadence Spectre 工具对振荡器进行大信号分析，提取小信号模型的参数，并用指定的准静态 MOS 管模型将电路线性化。其次，我们建立小信号电路的 GPDD 数据结构及 s 展开 BDD 结构，并利用时变主极点分析中，电路拓扑结构恒定不变这一特点，将一次建立好的数据结构进行重复的求值运算。再次，我们利用连续 Muller 迭代方法无冗余地求解主极点信息，快速得到振荡器的时变主极点。最后，我们利用 GPDD 求解敏感度信息的便利性，建立了对振荡器时变主极点根轨迹进

行小信号敏感度分析的算法，以期望得到主极点根轨迹相对于电路元件值的变化规律。这是一种与数值化仿真方法完全不同的分析流程，我们期望这一流程能够带来电路分析效率上的提升，并提供更多有用的电路优化信息。

第四章 符号化电路综合方法

4.1 问题的提出

从前面的讨论中，我们指出若电路拓扑结构未发生变化，则其小信号形式的拓扑结构亦未发生变化。如果我们对一个恒定不变的电路进行分析和优化，则用 GPDD 对电路进行仿真，可以带来一些便利。

但是，在实际的电路设计中，我们经常需要对不同拓扑结构的电路进行对比。比较常见的是对相似的电路进行比较分析。为了使得电路设计符合指标要求，我们可能需要在电路中添加一些元件，例如补偿电容、调零电阻等。或者我们希望简化电路分析，删去一些元件，以求得近似分析的结果。比较常见的是用理想运放去代替实际的放大电路，以快速地对电路性能进行分析。在这种情况下，我们就需要处理电路元件的删除和添加。

对于我们的振荡器来说，我们也需要对不同结构的振荡器作出性能的比较。这些振荡器相互之间可能相差一些偏置的电容或者电感。我们需要一种更加灵活的结构，能够快速分析相近电路的差异。

对于数值化分析方法来说，电路元件的删除和添加需要我们重新生成电路矩阵，以对修改后的电路进行分析。由于数值化方法建立矩阵的速度很快，因此比较不同电路不会带来过大的消耗。

但是对于符号化分析方法来说，电路拓扑结构的改变就意味着我们需要重新生成电路的符号化数据结构。我们知道，这是符号化分析中最耗时的一个步骤。在电路的符号顺序不佳的情况下，这一步骤需要指数级别复杂度的时间。这对于电路的分析是十分不利的。

因此，我们希望能够找到一种方法，在电路拓扑结构发生元件删除和添加的情况下，只需要在原来的符号化数据结构的基础上，进行小许改动和操作，就可以将其修改至与新的电路拓扑结构同步。

4.2 元件极限操作

我们先来考虑电路元件的删除。考虑 R/L/C/E/F/G/H 这七种线性电路所包含的元件。

对于 R/L/C 等阻抗和导纳元件来说，删除元件有两种方式。

- (1) 将元件所在支路断开。

(2) 将元件所在支路短路。

我们来考虑进行短路或者断开操作对应的元件大小值情况。




R 	short	$R = 0$	$\frac{1}{R} = \infty$
	open	$R = \infty$	$\frac{1}{R} = 0$
C 	short	$C = \infty$	$sC = \infty$
	open	$C = 0$	$sC = 0$
L 	short	$L = 0$	$\frac{1}{sL} = \infty$
	open	$L = \infty$	$\frac{1}{sL} = 0$

图 4-1 电阻、电容、电感进行短路、断开操作对应的元件值及导纳值
Figure 4-1 Symbol value and conductance of Resistor, Conductor and Inductor when shorted or opened

从图 4-1 中可以看到，对 R、L、C 三种元件进行短路或者断开操作时，相当于对元件及其相应的导纳取极限值。短路操作，导纳取无穷大。断开操作，导纳取零。我们不再用短路、断开表示这两类操作，而用导纳值的取值来表示这两类操作。短路定义为“无穷大”（INF）操作，而断开定义为“取零”（ZERO）操作。

同样地，我们定义 E/F/G/H 四类元件的 INF 和 ZERO 操作。

当 E/F/G/H 的值取为无穷大，亦即等效为理想运放（零阻器 Nullor）时，我们定义为对这四类元件做 INF 操作。

当 E/R/G/H 的值取为 0 时，控制端与受控端的联系被消除，我们定义为对这四类元件做 ZERO 操作。

我们用 p 来表示某个电路元件在 GPDD 中的值（R/L/C 取导纳值，E/F/G/H 取

本身值), 则从第二章的介绍中, 我们知道, 若 GPDD 定义了输入输出端口, 并将 p 作为在输入输出端口后的第一个符号 (symbol) 来进行处理, 则我们可以作出图 4-2。

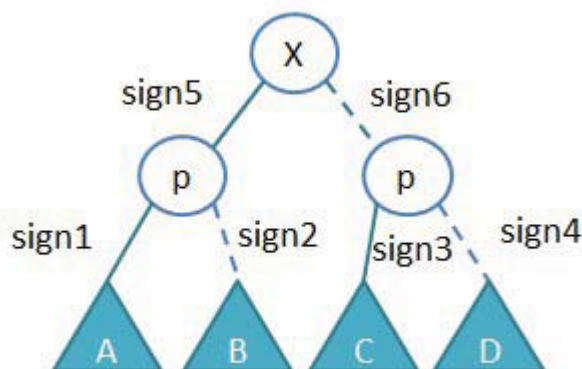


图 4-2 一个 GPDD 例子
Figure 4-2 An Example GPDD

实线表示 1-路径, 虚线表示 0-路径。系统函数可以表示为如下所示的形式。

$$H(s) = \frac{1}{X} = \frac{N(s)}{D(s)} = -\frac{\text{sign5} * (p * \text{sign1} * A + \text{sign2} * B)}{\text{sign6} * (p * \text{sign3} * C + \text{sign4} * D)} \quad (4-1)$$

其中, $N(s)$ 和 $D(s)$ 都是乘积项之和 (SoP) 的形式。 $p * A$ 表示 $N(s)$ 中包含 p 的乘积项之和, B 表示 $N(s)$ 中不包含 p 的乘积项之和。同理, $p * C$ 表示 $D(s)$ 中包含 p 的乘积项之和, D 表示 $D(s)$ 中不包含 p 的乘积项之和。

那么, 当 p 取 0 时, 我们有

$$H(s)|_{p=0} = \frac{N(s)|_{p=0}}{D(s)|_{p=0}} = -\frac{\text{sign5} * \text{sign2} * B}{\text{sign6} * \text{sign4} * D} \quad (4-2)$$

当 p 取无穷大时, 我们有

$$H(s)|_{p=\infty} = \frac{N(s)|_{p=\infty}}{D(s)|_{p=\infty}} = -\frac{\text{sign5} * \text{sign1} * A}{\text{sign6} * \text{sign3} * C} \quad (4-3)$$

通过对数学表达式的观察, 我们发现, 对 p 取 0 和对 p 取无穷大, 实际上是只取 $H(s)$ 的一部分并且忽略另外一部分。具体来说, 当 p 取 0 时, 我们需要求得 $H(s)$ 中所有不包含 p 的项做运算。当 p 取无穷大时, 我们需要求得 $H(s)$ 中所有包含 p 的项做运算。

那么, 式 (4-2) 和式 (4-3) 分别是图 4-2 中的一部分。分别用 GPDD 表示, 我们可以得到图 4-3。

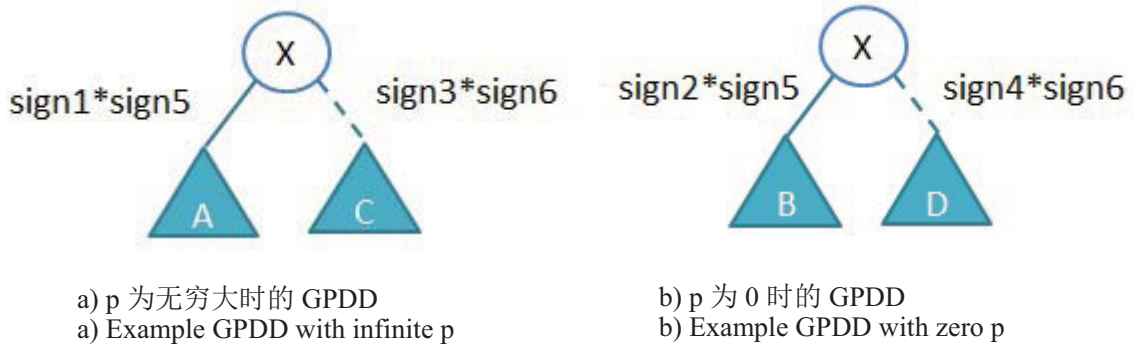


图 4-3 p 取极限值后的 GPDD 结构
Figure 4-3 GPDD with extreme p

我们可以从图中很清晰地看到，图 4-3 中的左右两个图都是图 4-2 中的局部。要从图 4-2 中得到图 4-3a) 中或图 4-3b) 中的系统函数，是很方便的。

要从图 4-2 中得到图 4-3a) 中的系统函数，我们只需要将 p 置为 1，并忽略 p 的 0-路径即可。

要从图 4-2 中得到图 4-3b) 中的系统函数，我们只需要将 p 置为 0，并忽略 p 的 1-路径即可。

在这个例子中， p 是作为输入输出端后面的第一个符号 (symbol) 来处理的。计算比较简单。

但实际上，在 GPDD 构建的过程中有一定的符号顺序 (order)， p 未必是在输入输出端后面的第一个符号。

为了比较不同的电路结构，我们可能需要同时对多个元件进行取极限操作，部分元件取 0，部分元件取无穷大。为了解决这个问题，我们需要一个更加通用的算法。

通过式 (4-2) 以及式 (4-3) 的分析，我们发现，如果含有一个无穷大的元件，那么分子和分母的 SOP 中的项必须包含这个元件，并且 SOP 中这个元件的值置为 1。

如果有一个为 0 的元件，那么分子和分母的 SOP 项中必须不包含这个元件。

对于多个元件为无穷大或者多个元件为 0 的计算，以上的准则可以进一步推广。我们余下的 SOP 中，必须包含所有无穷大的元件，不包含任意一个为 0 的元件。并且，SOP 中无穷大的元件值置为 1。我们用一个例子来说明这一计算过程。不考虑具体符号，我们考虑如下的 SOP。

$$ABCDE + ABDE + ACE + ADE + BCD + DE \quad (4-4)$$

在式 (4-4) 中，我们假设 A 与 D 为无穷大，B 为 0，C 与 E 为正常值。

根据式 (4-3) 和式 (4-3) 的分析, 我们知道, 我们需要留下的乘积项中必须不含有 B, 且必须包含 A 和 D。因此, 式 (4-4) 中符合条件的只有 ADE 这一项。

我们可以对式子进行循环二分分析。假设我们处理符号的顺序为 A->B->C->D->E, 则式子 (4-4) 首先可以化为包含 A 和不包含 A 的两部分

$$A(BCDE + BDE + CE + DE) + BCD + DE \quad (4-5)$$

由于 A 必须被包含, 因此我们可以舍去 $BCD + DE$ 这一项。将 A 置为 1, 进入括号内的下一层, 即

$$B(CDE + DE) + CE + DE \quad (4-6)$$

进一步根据 B 进行二分, 由于 B 为 0, 因此包含 B 的项需要被舍去, 剩下

$$C(E) + DE \quad (4-7)$$

接下来处理 C。C 为正常元素值, 所以所有的项都需要被考虑。先处理包含 C 的项。我们发现, 括号内的起始元素为 E, 跳过了 D, 因此需要被舍去。接下来, 我们处理不包含 C 的项 DE。对 D 进行二分, 我们发现只包含了含有 D 的项。

$$D(E) \quad (4-8)$$

将 D 置为 1, 进入最后一层 E, 结束我们的循环分析。

总结上述过程, 我们最后留下的就是项 ADE。将 A 和 D 置为 1 后, 此项变为 E。

二分过程, 实际上对应于 GPDD 结构走实线和走虚线的过程。

我们用一个 GPDD 结构来表示式 (4-4), 进一步阐释我们的算法。如图 4-4 所示。

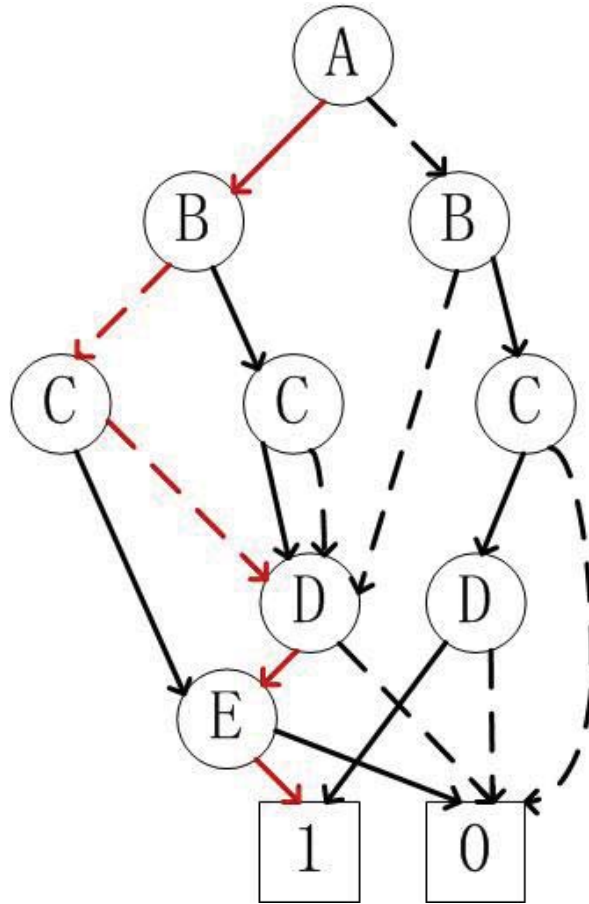


图 4-4 应用极限算法的一个例子
Figure 4-4 an example to apply infinity algorithm

图 4-4 中，实线依然表示乘法，虚线依然表示加法。为了简要地说明问题，我们在图中没有标示符号。图中红色的线就是我们需要寻找的乘积项 ADE 的路径。对代数表达式按含与不含某元素做二分操作，就对应于在 GPDD 中走 1-路径或者走 0-路径的操作。实际上，我们是按照无穷大与零元素筛选乘积项的准则，逐步在向下递归搜索过程中去除不符合条件的搜索路径，一步步缩小我们的搜索范围，并最终计算我们所需要的项的。

我们用一个 Slist 按顺序存放所有的符号，INFlist 表按顺序存放所有的无穷大的符号，并且用 nextINF 指向搜索过程中，下一个需要被包含的无穷大符号。数据结构如图 4-5 所示。

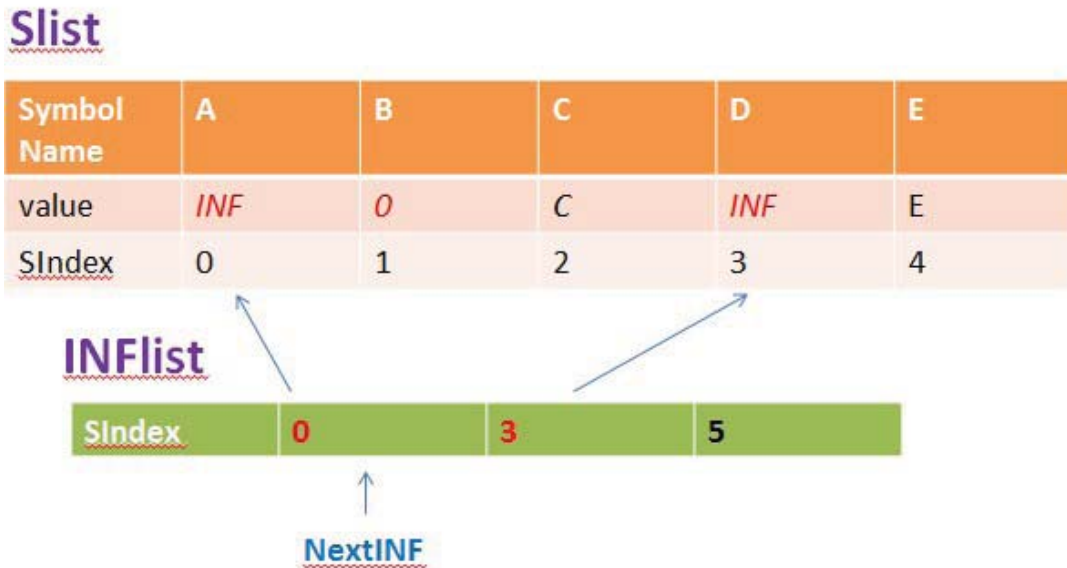


图 4-5 极限运算的数据结构
Figure 4-5 Data structure for infinity evaluation

其中，INFlist 的末端放置一个比 SIndex 最大值 4 大 1 的数 5，当 nextINF 指向这个数值时，不再有任何无穷大值需要被包含。

在 GPDD 中，ExpIndex 越小的符号在越高的层中，ExpIndex 越大的符号在越低的层中。我们定义搜索过程 $\text{eval}(\text{node} * P)$ ，用 P 表示当前的 GPDD 节点， $P \rightarrow \text{Left}$ 表示连接 P 的 1-路径上的点， $P \rightarrow \text{Right}$ 表示连接 P 的 0-路径上的点， sign1 表示 1-路径上的符号， sign0 表示 0-路径上的符号。 $S(P)$ 表示当前节点的导纳值，即图 4-5 中的 value， $V(P)$ 表示当前节点中，我们所需要计算的乘积项值。

则递归搜索算法的伪代码如下。

```

eval(P, nextINF)
{
  if (P->marked) return;
  else (mark(P)); //设置标记，避免重复计算
  if (S(P)==INF) //情况 1：当前节点为无穷大，只关心 1-路径。
  {
    //判断 1-路径上的点是否越过了下一个无穷大符号，如果未越过则递归进行计算，否则当前点的值为 0。
    if (P->Left->ExpIndex <= INFlist[nextINF+1])
      { eval(P->Left, nextINF+1); V(P)=sign1 * V(P->Left); }
    else
      { V(P)=0; }
  }
}

```

```

else if (S(P)==0) //情况 2: 当前节点为 0, 只关心 0-路径。
{
//判断 0-路径上的点是否越过了下一个无穷大符号, 如果未越过则递归进行计算, 否则当前点的值为 0。
if (P->Right->ExpIndex <= INFlist[nextINF])
{ eval(P->Right, nextINF); V(P)=sign0*V(P->Right);}
else
{ V(P)=0;}
}
else //情况 3: 当前节点为正常值, 0-路径及 1-路径都需要计算。
{
//判断每条路径上的点是否越过了下一个无穷大符号, 如果未越过则递归进行计算, 否则当前路径的值为 0。
if (P->Left->ExpIndex <= INFlist[nextINF])
{ eval(P->Left, nextINF); V_left=V(P->Left); }
else
{ V_left=0; }
if (P->Right->ExpIndex <= INFlist[nextINF])
{ eval(P->Right, nextINF); V_right=V(P->Right);}
else
{ V_right=0;}
//最后计算当前点的值
V(P) = S(P)*sign1*V_left+sign0*V_right;
}
}
}

```

我们通过图 4-4 来说明算法是如何运作的。为了简洁, 我们假设所有路径上的符号均为正。

首先, 我们需要探索的无穷大点为 A。

从结构的顶点开始搜索。

- (1) 首先遇到无穷大的项 A, 符合情况 1, 将 nextINF 指向点 D, 走 1-路径。
- (2) 到达 B 点, B 为 0, 符合情况 2。0-路径指向 C, 未越过 D, 合法。走 0-路径到达 C。
- (3) C 为正常项, 符合情况 3。C 的 1-路径指向 E 点, 越过了 D, 因而这条路径上的乘积项必然不包含 D。此路径不合法。V_left 设为 0。C 的 0-路径指向 D, 合法。走 0-路径到达 D。

- (4) D 为无穷大点, 符合情况 1, 将 nextINF 设为比最底层 E 的需要大于 1 的数, 表示在遍历 D 以后的点时, 无需再用无穷大的条件判断路径的合法性。由于 D 为无穷大点, 我们走 1-路径到达 E 点。
- (5) E 连接终端, 可以计算 $V(E) = E * 1 + 0 = E$ 。这时, 递归计算可以逐步回溯。回到点 D, D 为无穷大点, 置为 1 并且计算实线端的值, $V(D) = 1 * V(E) = E$ 。回到 C 点, 实路径不合法, 因此 $V(C) = C * 0 + V(D) = E$ 。回到 B 点, $V(B) = V(C) = E$ 。回到 A 点, $V(A) = 1 * V(B) = E$ 。

这样, 我们就完成了对图 4-4 的遍历及运算, 得到了正确的结果。

需要注意的是, 如果电路中有多个并联的元件同时为无穷大, 则由 GPDD 得到的 SOP 无法同时包含这些无穷大元件。由于多个并联的无穷大元件与一个无穷大元件是等效的, 我们在进行计算之前, 做一个预处理操作, 把它们归并为一个无穷大元件来处理就可以了。

4.3 元件添加操作

我们已经讲述了对电路中的元件进行极限化 (置零或者无穷大) 后, 如何在已有的 GPDD 结构中进行自顶向下的逐步筛选和递归运算, 得到正确的结果。

这是在一个完整 GPDD 的基础上, 计算局部 GPDD 的运算。元件进行极限化操作, 是对电路进行简化。

但是, 在进行电路设计时, 我们经常需要将电路复杂化。可能需要在一些支路上串联添加一个元件, 也可能需要选择一些节点, 并联地添加元件。

我们也可能需要将一些理想化的元件 (如理想运放) 转换为实际的放大器, 或者将一些简单的电路模型复杂化, 添加电路不同分支之间的控制关系。

不同于数值化的矩阵映射关系, 用符号化方法分析电路时, 我们需要建立与电路相对应的符号化数据结构。这个数据结构和电路的拓扑结构是相互对应的。在过去, 电路结构的更改就意味着整个符号化结构的重建, 这是十分耗时的。在高频振荡器的分析中, 我们需要对不同结构的振荡器进行性能的比较。若每次进行比较都需要重建结构, 则会大大降低符号化分析的效率。

为了解决这个问题, 我们提出一套符号化的元件添加算法, 在不删除和重构 GPDD 的情况下, 通过对 GPDD 结构作一些修改, 使之能动态地适应线性元件的添加。

当我们提到“添加”这一概念时，对 R/L/C 元件，指的是以下的一些操作。

- (1) 在原电路的一个节点撕裂为两个节点，并接入元件。
- (2) 加入元件，连接原电路的两个不同节点。

图 4-6 是操作 (1) 的示意图。图 4-7 是操作 (2) 的示意图。

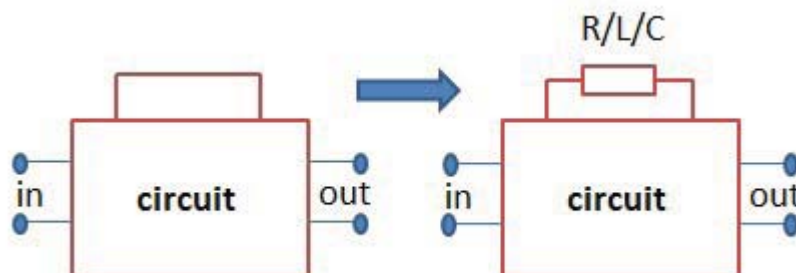


图 4-6 撕裂节点并接入元件示意图

Figure 4-6 Sketch map of splitting a node and inserting a device

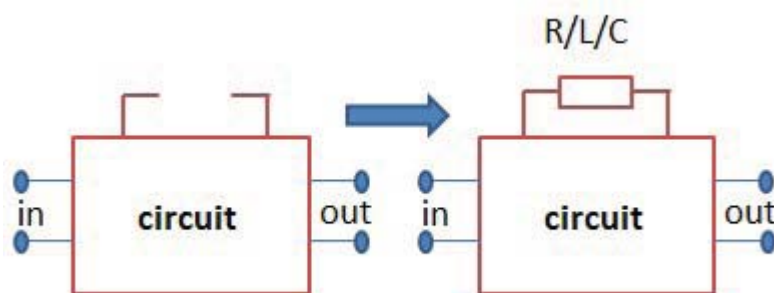


图 4-7 接入元件连接不同节点示意图

Figure 4-7 Sketch map of inserting device to connect two separate nodes

实际上，图 4-6 中，R/L/C 这一条支路是由一个无穷大的导纳变为一个具体的导纳值，而图 4-7 中，R/L/C 这一条支路是由一个 0 导纳值变为一个具体的导纳值。

我们将添加操作扩展到 E/F/G/H 元件，将由理想运放 (Nullor) 转化到具体 E/F/G/H 值的变化，以及将元件值为 0 的情况转化为具体 E/F/G/H 值的变化定义为对 E/F/G/H 元件的两类不同的添加方法。

回顾 4.3 节，我们可以很容易地对电路进行元件从具体值变为极限值的操作。实际上，针对电路中的任意一个元件 p ，我们可以将系统函数 $H(s)$ 写成下面的形式。

$$H(s) = \frac{N(s)}{D(s)} = \frac{p * N(s)|_{p=\infty} + N(s)|_{p=0}}{p * D(s)|_{p=\infty} + D(s)|_{p=0}} \quad (4-9)$$

其中， $p * N(s)|_{p=\infty}$ 定义为 $N(s)$ 中含有 p 的乘积项之和 (SOP)。 $N(s)|_{p=\infty}$ 为这

个 SOP 除以 p 的结果。 $N(s)|_{p=0}$ 定义为 $N(s)$ 中所有不含 p 的乘积项之和。 $D(s)|_{p=0}$ 与 $D(s)|_{p=\infty}$ 的定义类似。我们有

$$H(s)|_{p=\infty} = \frac{N(s)|_{p=\infty}}{D(s)|_{p=\infty}} \quad (4-10)$$

以及

$$H(s)|_{p=0} = \frac{N(s)|_{p=0}}{D(s)|_{p=0}} \quad (4-11)$$

我们现在可以将问题转化为，在预先已经构造好知道 $H(s)|_{p=\infty}$ 的 GPDD 或者 $H(s)|_{p=0}$ 的 GPDD 的情况下，如何通过较少的改动得到 $H(s)$ 对应的 GPDD。

以图 4-6 为例，实际上，我们可以通过第二章所介绍的 GPDD 的构造规则，将图 4-6 的右图转化为左图。这里包含了一个对 R/L/C 元件的预处理操作。根据 GPDD 的构造规则，我们可以将图 4-6 右图中的电路拆分为 GPDD 子图对，然后对 R/L/C 元件做包含 (Include) 操作。这个操作所得到的结果与图 4-6 左图电路对应的 GPDD 子图对是一致的。

类似地，我们可以将图 4-7 中的右图转化为 GPDD 子图对，通过对 R/L/C 元件做移除 (Exclude) 操作，得到图 4-7 中的左图所对应的 GPDD 子图对。

同样，若我们将 E/F/G/H 元件取为无穷大值，即转化为 Nullor，并对 Nullor 在 GPDD 图对做预处理操作，可以得到一对子图对。这对子图对，与直接对 E/F/G/H 元件在 GPDD 图对中进行 Include 操作所余下的子图对，是一致的。因此，将 E/F/G/H 取为无穷大值，相当于对 GPDD 子图对中的 E/F/G/H 元件预先进行 Include 操作。

若将 E/F/G/H 元件置为 0，则其等效电路图的 GPDD 子图对，与将 E/F/G/H 元件在 GPDD 子图对做 Exclude 操作得到的子图对，是一致的。

总结以上的结论，假设电路中有一个元件 p ，则我们可以得到两条规则：

(1) 预先对元件 p 做 Include 操作，得到一个子图对。对该子图对进行 GPDD 展开，所得到的 GPDD，与 $H(s)|_{p=\infty}$ 一致。

(2) 预先对元件 p 做 Exclude 操作，得到一个子图对。对该子图对进行 GPDD 展开，所得到的 GPDD，与 $H(s)|_{p=0}$ 一致。

我们假设 $H(s)|_{p=\infty}$ 与 $H(s)|_{p=0}$ 的结构如图 4-8 所示。

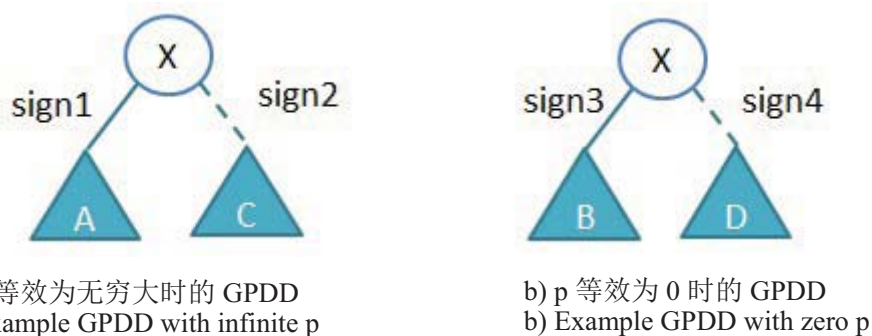


图 4-8 不包含 p 的原电路 GPDD 结构
Figure 4-8 GPDD of the original circuit without p

现在，我们在电路中加入元件 p ，并且将 p 作为第一个符号对电路做 GPDD 展开。根据前面得到的两条规则，可以得到图 4-9。

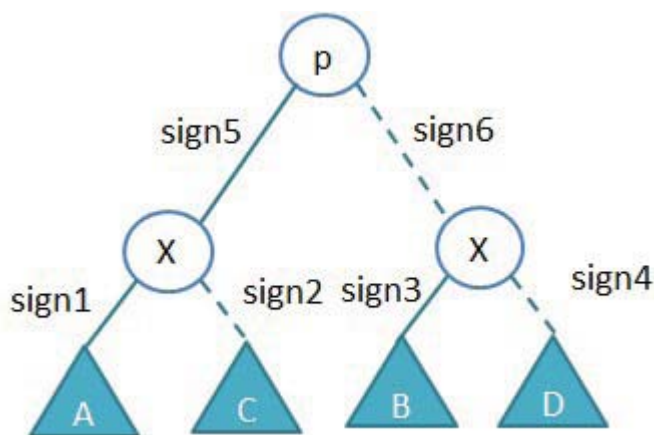


图 4-9 添加 p 后的 GPDD 结构
Figure 4-9 GPDD of the circuit with p added

其中， p 的 1-路径指向图 4-8a) 中的结构，0-路径指向图 4-8b) 中的结构。

我们来分析从根节点的有向子图对，到 A, B, C, D 四个子 GPDD 的有向子图对，需要经过哪些操作。

从根节点到 A，经历了两步的操作：Include p ，Include X。

从根节点到 C，经历了两步的操作：Include p ，Exclude X。

从根节点到 B，经历了两步的操作：Exclude p ，Include X。

从根节点到 D，经历了两步的操作：Exclude p ，Exclude X。

对应于第二章中的 GPDD 展开规则，我们知道，每一个 Include 操作和 Exclude 操作都对应于电路中某些边的断开或者短路。我们来看第一步操作。

从 p 到 A，需要先 Include P ，再 Include X。如果操作相反，先 Include X，再 Include P ，结果会如何呢？实际上，无论操作的顺序如何，最终产生的被短路的

边的集合以及被开路的边的集合都是一致的。我们可以很容易地证明，如果对同一个集合中的元件进行 Include 或 Exclude 操作，则无论操作顺序如何，最终得到的 GPDD 子图对都是一样的。因此，我们可以更改一下操作顺序。

从根节点开始，经历两步的操作：Include X，Include p 后，可以得到点 A。

从根节点开始，经历两步的操作：Include X，Exclude p 后，可以得到点 B。

从根节点开始，经历两步的操作：Exclude X，Include p 后，可以得到点 C。

从根节点开始，经历两步的操作：Exclude X，Exclude p 后，可以得到点 D。

也就是说，如果我们调整 GPDD 的展开顺序，先对 X 做展开，再对 p 做展开，那么我们可以得到如图 4-10 所示的数据结构。

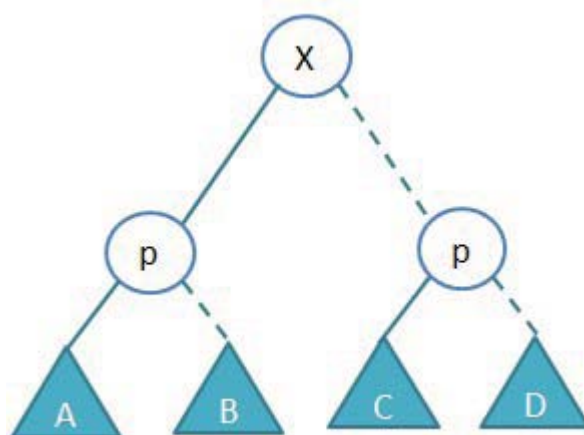


图 4-10 调整后的包含 p 的电路的 GPDD 结构
Figure 4-10 Reordered GPDD of the circuit with p added

图 4-10 中，A、B、C、D 四个子 GPDD 与图 4-9 中的 A、B、C、D 这四个子 GPDD 是完全一致的，只是在 GPDD 中的位置有所不同。由于图 4-9 中的 A、C 两个子 GPDD 与图 4-8a) 中一致，图 4-9 中的 B、D 两个子 GPDD 与图 4-8b) 中亦一致。因此，如果我们已经有图 4-8a) 中的数据结构，需要插入元件 p 时，可以将 A、C 放到 4-10 的对应位置中，并构造 B、D 两个子 GPDD。如果我们已经有图 4-8b) 中的数据结构，需要插入元件 p 时，可以将 B、D 放到 4-10 的对应位置中，并构造 A、C 两个子 GPDD。

我们用图 4-11 与图 4-12 来详细说明这两个过程。

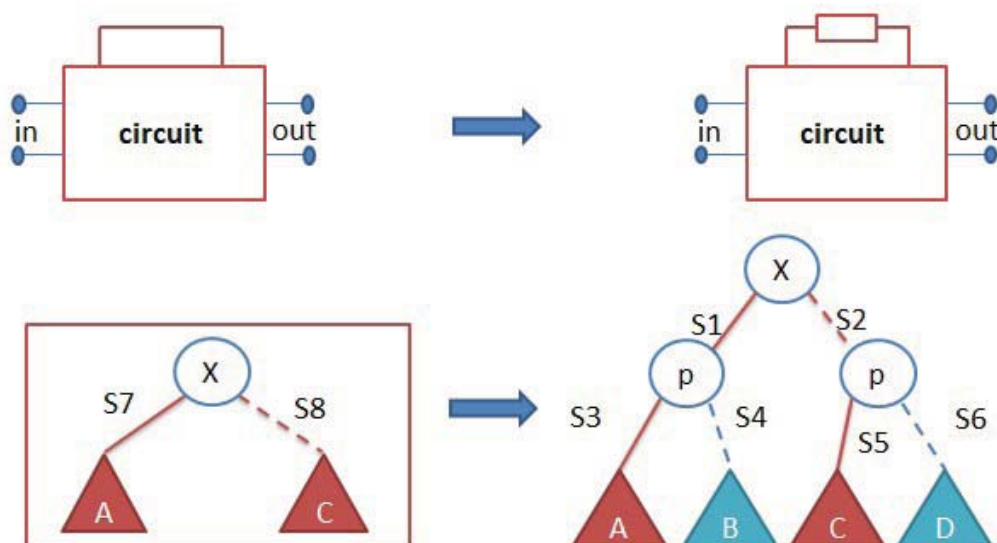


图 4-11 元件 p 由无穷大变化到具体值的 GPDD 变化过程
Figure 4-11 Change on GPDD when p changes from infinity to a concrete value

若元件 p 为 R/L/C，则图 4-11 表示撕裂原电路中的一个节点，并在其中插入元件 p ，得到新电路的过程。

若元件 p 为 E/F/G/H，则图 4-11 表示将原电路中的一个理想运放（Nullor）替换为有限增益放大器的过程。

在这两个过程中， p 由无穷大的导纳或者放大倍数，变化到有限值。

原电路中已包含 A、C 两项，我们需要在新电路中生成 B、D 两项，并且决定 S1, S2, S3, S4, S5, S6 这六个符号的值。具体过程如下。

- (1) 删除原电路中的 GPDD 根节点 X。
- (2) 根据新电路，生成新的 GPDD 根节点 X 及其所对应的两个子图。
- (3) 对 X 进行 Include 和 Exclude 操作，得到 S1 和 S2 的值及两个 p 节点。
- (4) 对两个 p 节点分别进行 Include 操作，得到 S3 及 S5，并将连接 p 节点的 1-路径分别指向原电路图中的 A 和 C 两个子 GPDD。
- (5) 对两个 p 节点分别进行 Exclude 操作，得到 S4, S6 以及 B、D 两个子 GPDD 的根节点。
- (6) 自顶向下构造 B 及 D 两个子 GPDD。

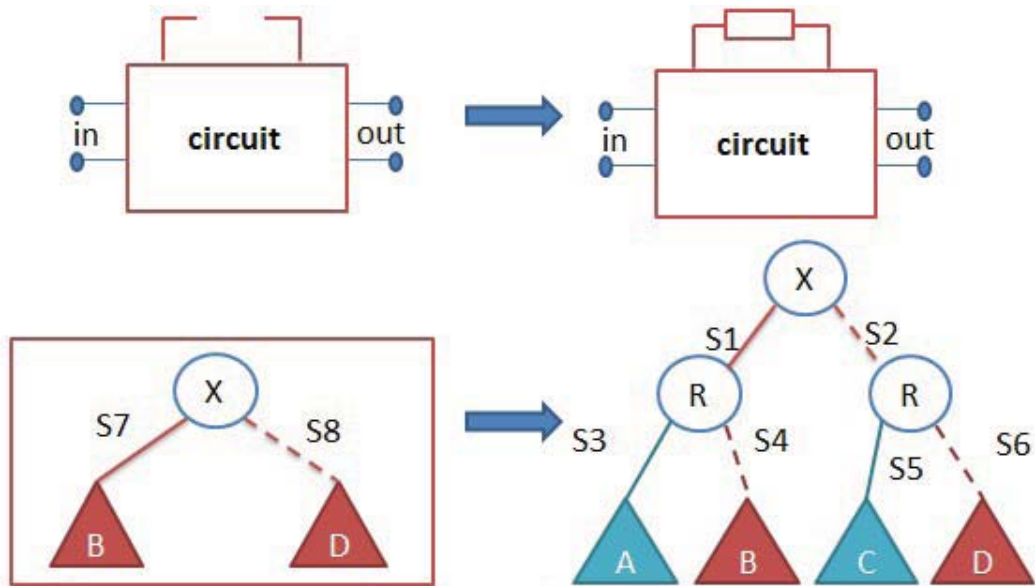


图 4-12 元件 p 由 0 变化到具体值的 GPDD 变化过程
Figure 4-12 Change on GPDD when p changes from infinity to a concrete value

若元件 p 为 R/L/C，则图 4-12 表示在原电路的两个节点中，插入元件 p ，得到新电路的过程。

若元件 p 为 E/F/G/H，则图 4-12 表示在原电路中新增受控源 p ，得到新电路的过程。

在这两个过程中， p 由为 0 的导纳值或者放大倍数，变化到一个具体数值。

原电路中已包含 B、D 两项，我们需要在新电路中生成 B、D 两项，并且决定 S1, S2, S3, S4, S5, S6 这六个符号的值。具体过程如下。

- (7) 删除原电路中的 GPDD 根节点 X。
- (8) 根据新电路，生成新的 GPDD 根节点 X 及其所对应的两个子图。
- (9) 对 X 进行 Include 和 Exclude 操作，得到 S1 和 S2 的值及两个 p 节点。
- (10) 对两个 p 节点分别进行 Exclude 操作，得到 S4 及 S6，并将连接 p 节点的 0-路径分别指向原电路图中的 B 和 D 两个子 GPDD。
- (11) 对两个 p 节点分别进行 Include 操作，得到 S3, S5 以及 A、C 两个子 GPDD 的根节点。
- (12) 自顶向下构造 A 及 C 两个子 GPDD。

这样，我们便完成了在完整利用原 GPDD 的情况下，添加元件 p ，构造包含元件 p 的新 GPDD 的过程。

在实现时，我们需要注意一些地方。对一个电路的 GPDD 进行构造时，我们通常需要进行下列几个步骤。

- (1) 初始化 GPDD，生成根节点。
- (2) 自顶向下进行 GPDD 构造，同时根据图的哈希以及图对的哈希加速构造过程，共享内部节点。
- (3) 自底向上进行 Reduction 和 Zero Suppress 过程。使 GPDD 变得更加紧凑。

我们从图 4-10 及图 4-11 中可以看出，新添加的子 GPDD 是自顶向下构造的。为了最大化这个构造过程，则需要使图的哈希以及图对的哈希索引命中率最大化。

图对的哈希，键值与返回值均为 GPDDNode 类，输入一个 GPDD 节点，返回与之具有相同图对的哈希表中的节点。

在[18]的构造过程中，进行步骤(2)后即会将 GPDD 中图的哈希及图对的哈希删除，以释放内存空间。为了使 GPDD 元件添加效率最大化，我们不删除这些哈希结构。

在[18]的构造过程中，进行 Reduction 和 Zero Suppress 时，需要对 GPDD 中的指针进行重定向，并删除一些节点。在这里，我们做对 GPDD 中的指针进行重定向的操作，而不删除这些 GPDD 节点，将它们仍然保留在图对哈希结构中。

我们对这些原本需要被删除的节点做标记，对被 Reduction 操作删除的节点添加一个 Rref 指针，使其指向紧凑结构中的点。

对被 Zero Suppress 的节点，我们添加一个 Zsupressed 标志位，标记此节点需要被删除。

当我们在构建新的子 GPDD 过程中，GPDD 节点被图对哈希表哈希到一个 GPDDNode 时，可以停止向下构造。我们进行如下的操作。

- (1) 若当前所引导的 GPDDNode 中的 Zsupressed 标志位为真，则进入(2)。否则进入(3)。
- (2) 当前指针指向 GPDDNode 0-路径指向的 GPDD 节点，并更新当前指针的符号。若新指向的 GPDD 节点 Zsupressed 位依然为真，则重复(1)操作，否则进入(2)。
- (3) 若当前索引到的 GPDDNode 所对应的 Rref 不为空，则当前指针指向 Rref。结束。否则进入(4)。
- (4) 当前指针指向索引到的 GPDDNode。结束。

通过这些考虑，新 GPDD 构造过程中的效率能够得到提升。

综上所述，我们给出了将元件 p 插入电路中时，GPDD 结构进行相应变化的

算法。我们可以处理以下几种情况。

- (1) 元件 p 为 $R/L/C$, 可以撕裂原电路中的一个节点, 并在其中插入元件 p 。也可以在原电路已有的两个节点中, 插入元件 p 。
- (2) 若元件 p 为 $E/F/G/H$, 可以将原电路中的一个理想运放 (Nullor) 替换为有限增益的 p 。也可以在原电路中新添加受控源 p 。

这一算法是非常灵活的。如果电路中所添加的元件不止一个, 那么, 我们可以用相同的算法逐一添加元件。每次添加一个元件, GPDD 结构增加一个新的元件层。

4.4 本章小结

本章详细地讨论了在电路拓扑结构发生变化时, 即对元件进行添加和删除时, 如何最大化地利用已有的符号化 GPDD 结构中的信息, 对 GPDD 进行动态的修改和更新, 使之能够处理 $R/C/L/E/F/G/H$ 这七种元件从有限值到极限值, 或从极限值到有限值的变化。

利用本章中提出的算法, 能够给 GPDD 较大的灵活度。在对相似的电路结构进行比较时, 无需对 GPDD 进行重复构造, 只需要对 GPDD 做相应的省略或补充即可。

为了通用性, 本章提出的算法适应于有输入输出的电路。实际上, 自激振荡电路也可以用类似的算法进行比较分析。当利用这一算法对振荡电路的特征多项式进行分析时, GPDD 的构造和求值只需考虑根节点引出的 0-路径下的 GPDD 结构, 其他部分, 包括根节点以及从根节点引出的 1-路径下的 GPDD 结构, 均不需要考虑。

第五章 实验结果分析

5.1 符号化仿真器与数值化仿真器的对比分析

本文所述的仿真器由 C++ 编写，在 Linux 环境下进行编译测试。所有测试结果均在个人电脑上测得。具体测试环境如下：

CPU: Intel Core2 2.26GHz

内存: 256MB

操作系统: Linux

为了对比符号化仿真器的正确性和效率，我们用 C++ 程序调用 LAPACK QZ 过程得到数值方法计算的结果，与符号化仿真器的结果作对比分析。

我们用第三章中介绍的流程实现符号化仿真器。

实验中，振荡器的 PSS 求解结果用 Cadence Spectre 工具仿真得到，MOS 管用 TSMC 0.18um 的模型库进行仿真，用图 3-3 中的准静态模型将其线性化。

数值化仿真器的大信号分析部分与符号化方法类似。

小信号分析部分，符号化仿真器的计算主要分为三个过程：

- (1) 构造：根据小信号网表的拓扑结构，构建符号化 GPDD 结构，并在其基础上构造 s 展开 BDD。
- (2) 计算系数：在时间点 t_k ，读入各项小信号参数值，更新 GPDD 符号表，深度优先遍历 s 展开 BDD 结构，计算 $D(s, t_k)$ 各项系数值。
- (3) 主极点求解：基于 $D(s, t_k)$ 的系数值，用连续 Muller 求根方法求解 t_k 上的主极点。

若 PSS 包含的采样时间点有 n 个，则过程 (2) 与过程 (3) 重复 n 次。

数值化仿真器在小信号分析部分主要分为两个过程：

- (1) 在时间点 t_k ，读入小信号参数值，构造 $D(s, t_k)$ 对应的 $C(t_k)$ 与 $G(t_k)$ 两个矩阵。
- (2) 用 LAPACK QZ 分析过程基于 $C(t_k)$ 与 $G(t_k)$ 两个矩阵求解 $D(s, t_k)$ 的所有根，并从中挑选 t_k 上的主极点。

下面我们通过一个交叉耦合振荡器和一个带偏置的交叉耦合振荡器对仿真器的性能做分析。

5.1.1 测试电路 1：交叉耦合振荡器

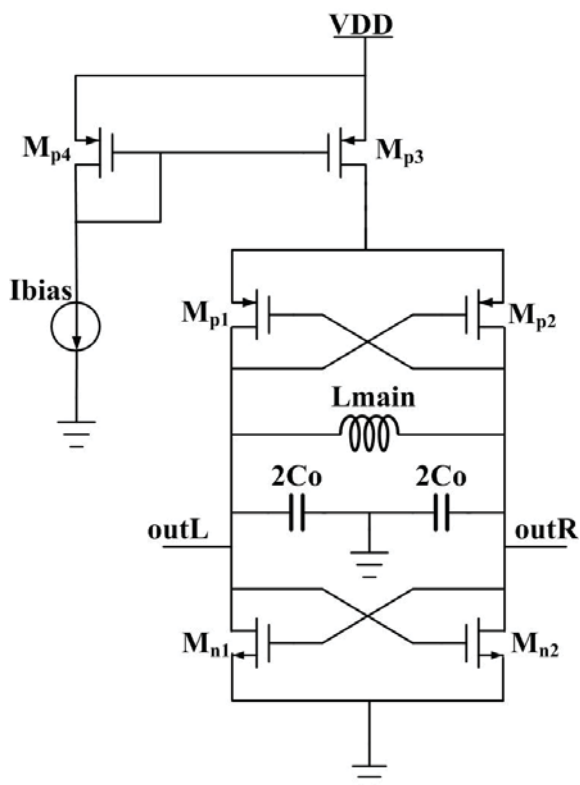


图 5-1 交叉耦合振荡器
Figure 5-1 Cross-coupled oscillator

图 5-1 中的各项参数如表 5-1 所示。

表 5-1 交叉耦合振荡器的各项参数值
Table 5-1 Parameter value of the cross-coupled oscillator

电路元件	Ibias	VDD	M _{p1} M _{p2} (W/L)	M _{n1} M _{n2} (W/L)
参数值	3.8mA	2.2V	60um/180nm	60um/180nm
电路元件	M _{p3} (W/L)	M _{p4} (W/L)	Lmain	Co
参数值	40um/180nm	40um/180nm	870pH	1.43pF

其中，Co 为理想电容，Lmain 用 TSMC 0.18um 中的 π 模型库构建。

PSS 输出结果振幅为 1V。

我们用数值方法和符号化方法得到的主极点根轨迹如图 5-2 所示。

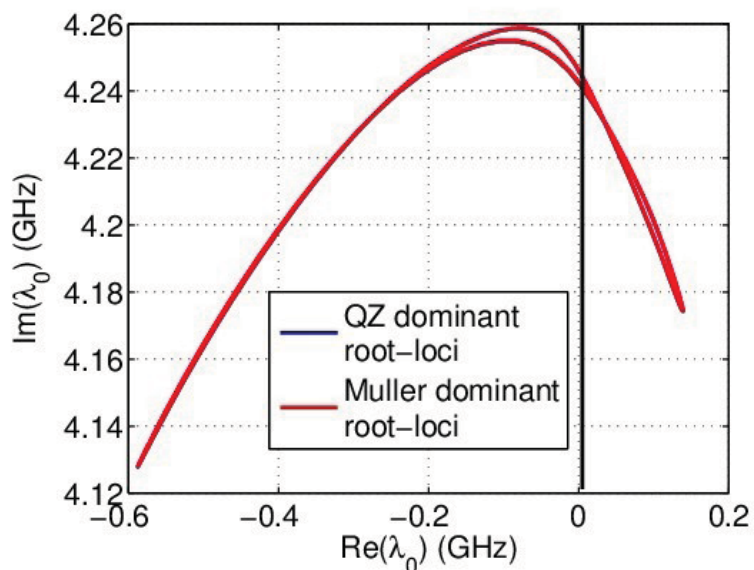


图 5-2 交叉耦合振荡器的主极点根轨迹
Figure 5-2 Main root-locus of cross-coupled oscillator

图 5-2 中，蓝线代表数值方法求得的主极点根轨迹，红线代表用符号化方法求得的主极点根轨迹。可见，两条轨迹是完全重合的。

对用两种方法求得的主极点数值做对比分析，我们发现，两种方法求得的主极点数值在 10 位有效数字上完全吻合。

图 5-3 中给出了用符号化方法求得的主极点根轨迹的实数部分与 PSS 分析的对照结果。

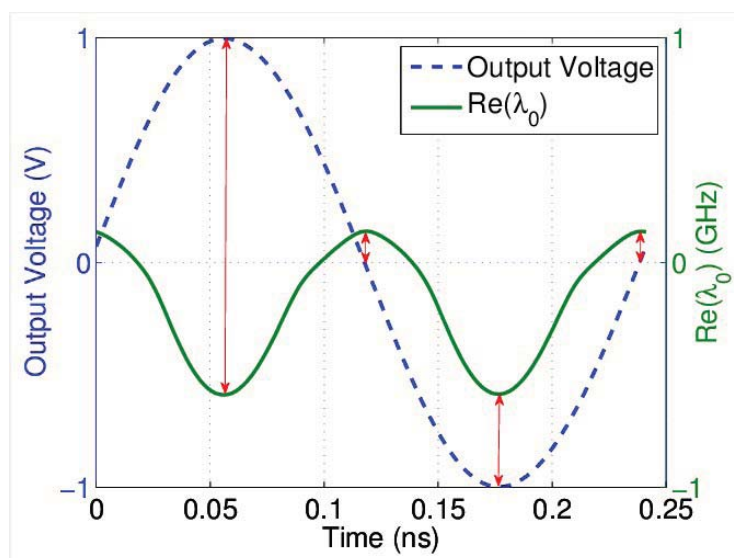


图 5-3 交叉耦合振荡器主极点根轨迹实数部分与 PSS 分析结果对比图
Figure 5-3 Main root-locus of the cross-coupled oscillator in comparison to PSS result

分析图 5-3，我们发现，图 5-2 中的根轨迹在一个振荡周期内共穿越了四次虚轴。当轨迹第一次到达复平面的最左值（根轨迹的实数部分最小）时， $V_{out}=1V$ ， M_{p1} 与 M_{n2} 处于线性区， M_{p2} 与 M_{n1} 截止，四个 MOS 管贡献正导纳。当轨迹第一次达到复平面的最右值（根轨迹的实数部分最大）时， $V_{out}=0V$ ， M_{p1} ， M_{n2} ， M_{p2} 与 M_{n1} 处于饱和区，贡献负导纳。当轨迹第二次到达复平面的最左值（根轨迹的实数部分最小）时， $V_{out}=-1V$ ， M_{p2} 与 M_{n1} 处于线性区， M_{p1} 与 M_{n2} 截止，四个 MOS 管贡献正导纳。当轨迹第一次达到复平面的最右值（根轨迹的实数部分最大）时， $V_{out}=0V$ ， M_{p1} ， M_{n2} ， M_{p2} 与 M_{n1} 再次处于饱和区，贡献负导纳。这一结果与理论相符，验证了符号化方法分析根轨迹的正确性。

5.1.2 测试电路 2：带偏置电感的交叉耦合振荡器

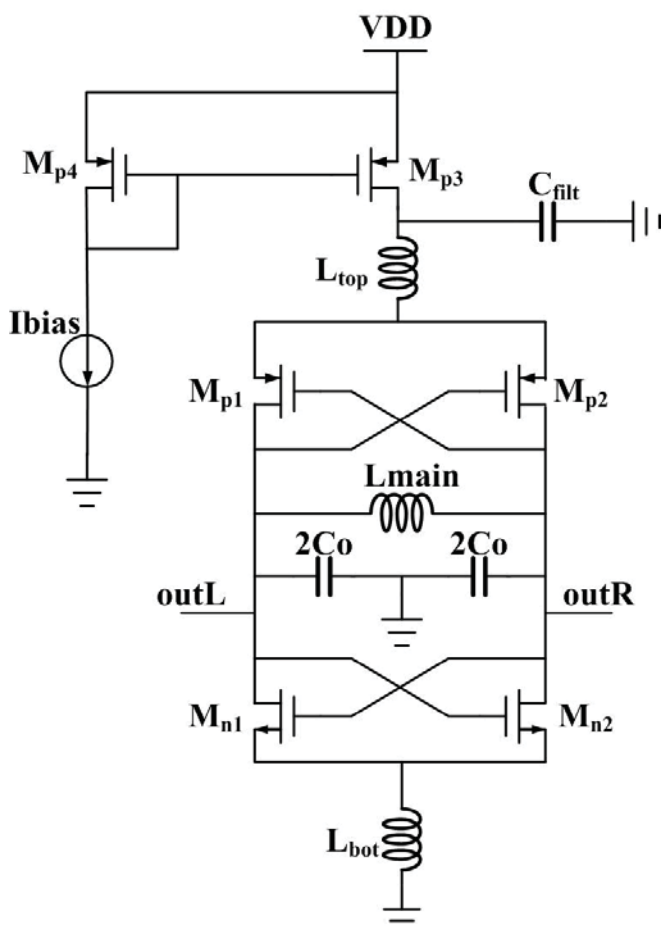


图 5-4 带偏置电感的交叉耦合振荡器
Figure 5-4 Cross-coupled oscillator with two biased inductors

图 5-4 中的各项参数如表 5-2 所示。

表 5-2 带偏置电感的交叉耦合振荡器的各项参数值
Table 5-2 The parameters of the cross-coupled oscillator with biased inductors

电路元件	Ibias	VDD	M_{p1} M_{p2} (W/L)	M_{n1} M_{n2} (W/L)
参数值	4.4mA	2.2V	60um/180nm	60um/180nm
电路元件	M_{p3} (W/L)	M_{p4} (W/L)	Lmain	Co
参数值	40um/180nm	40um/180nm	870pH	1.43pF
电路元件	L_{top}	L_{bot}	C_{filt}	
参数值	0.8nH	1.8nH	20pF	

其中, Co 为理想电容, Lmain 用 TSMC 0.18um 中的 π 模型库构建。L_{top} 与 L_{bot} 均为理想电感。

PSS 输出结果振幅为 1V。

我们用数值方法和符号化方法得到的主极点根轨迹如图 5-5 所示。

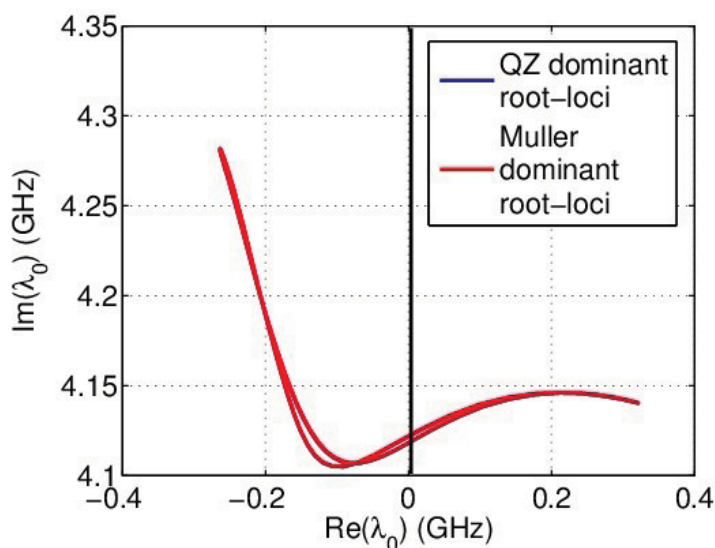


图 5-5 带偏置电感的交叉耦合振荡器的主极点根轨迹
Figure 5-5 Main root-locus of cross-coupled oscillator with biased inductors

从图 5-5 中可以看出, 用两种方法求得的主极点根轨迹是完全重合的。

对具体数值做分析, 两种方法求得的主极点数值在 10 位有效数字上完全吻合。

图 5-6 中给出了用符号化方法求得的主极点根轨迹的实数部分与 PSS 分析的

对照结果。

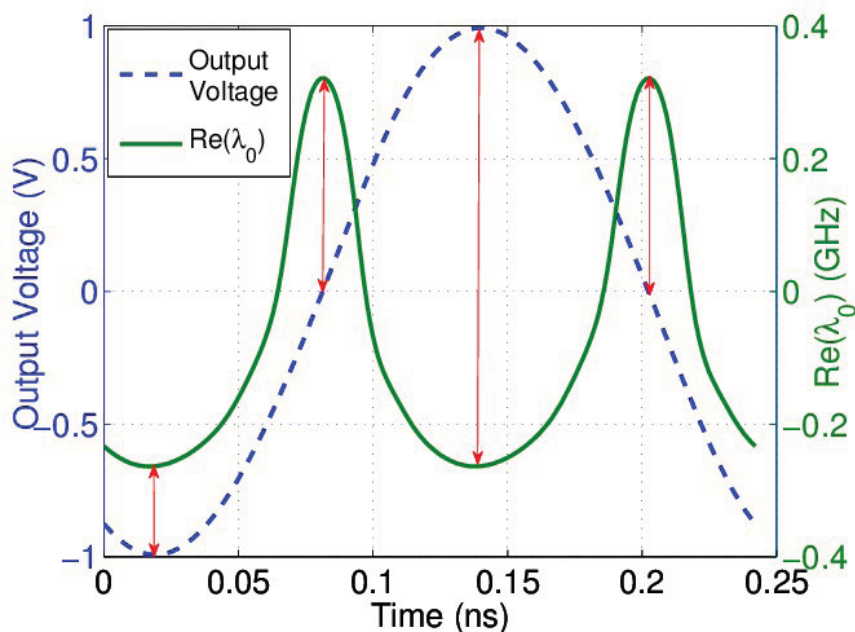


图 5-6 带偏置电感的交叉耦合振荡器主极点根轨迹实数部分与 PSS 分析结果对比图
Figure 5-6 Main root-locus of the cross-coupled oscillator with biased inductors in comparison to PSS result

分析图 5-6, 我们发现, $V_{out}=1V$ 或者 $V_{out}=-1V$ 时, 一对 MOS 管处于饱和区, 一对 MOS 管处于线性区, 四个 MOS 管均贡献正导纳, 使得根轨迹达到复平面的最左值。当 $V_{out}=0V$ 时, 四个 MOS 管贡献负导纳, 根轨迹达到复平面的最右值。这一结果与理论相符, 再次验证了符号化方法分析根轨迹的正确性。

5.1.3 符号化高频电路主极点根轨迹算法的效率分析

我们将符号化分析方法与数值化分析方法的效率比较统计于表 5-3。将交叉耦合振荡器表示为 Osc-1, 将带偏置电感的交叉耦合振荡器表示为 Osc-2。

两个电路的 PSS 采样点均为 202 个。

表 5-3 中不包含符号化数据结构的建立时间, 只统计计算系数的时间以及通过 Muller 迭代求根的时间。数值化分析方法统计 QZ 求根及筛选主极点的时间。

表 5-3 符号化分析方法与 QZ 数值化分析方法的效率比较
Table 5-3 Performance comparison between the symbolic method and the QZ method

	符号化方法			QZ 方法
	系数计算时间 (s)	Muller 主极点求解时间 (s)	总时间 (s)	求解及筛选主极点总时间 (s)
Osc-1 (12 个根)	0.058	0.006	0.064	0.141
Osc-2 (16 个根)	0.215	0.009	0.224	0.260

可以从表 5-3 中看出, 符号化方法求解主极点的总时间少于 QZ 方法计算的总时间, 在分析两个测试电路时效率比 QZ 方法要高。

用 QZ 方法分析两个振荡器电路, 所建立的矩阵大小分别为 16 阶及 20 阶。由于 QZ 方法需要计算使特征多项式为 0 的所有根, 并从中筛选主极点, 包含了计算冗余, 因此效率较低。

而符号化的分析方法基于多项式进行计算, 使用了连续 Muller 迭代的方法, 利用前一时刻点的结果作为初始值, 得到后一时刻点的结果, 这使得迭代速度得到了很大的提升。经过观察, 我们发现 Muller 迭代可在 4 到 5 个循环中达到收敛值, 因此, 我们可以在表 5-3 中看出, 这一过程的效率是非常高的, 在总计算时间中几乎可以忽略不计。

但是, 系数计算过程在符号化分析方法中占用了大部分的时间。这一时间与符号化计算的数据结构大小有关。我们在表 5-4 中给出具体的分析。

表 5-4 符号化方法的细节统计
Table 5-4 Details of the symbolic method

	符号数	s 展开 BDD 大小	GPDD 及 s 展开 BDD 构造总时间	系数计算时间 (s)
Osc-1	69	4088	0.20	0.058
Osc-2	84	10675	0.41	0.215

从表 5-4 中的数据可以看出, 振荡器对应的 s 展开 BDD 是非常大的。在符号顺序排列较差的情况下, 这一大小可能与符号数呈指数级别的对应关系。而系数计算时间则正比于 s 展开 BDD 的大小与所要分析的时间点个数的乘积。

目前有两种方法来改善这一部分的效率。一种是采用启发式算法, 使用更优的符号排列顺序^[18]以降低 GPDD 结构的大小。一种是用目前比较流行的基于 GPU 的符号化并行计算方法, 将时间复杂度降低到正比于符号数的级别^[31]。

若电路规模继续上升, 则需要采用层次化的方法对电路进行分析^[32]。但是对

于通常分析所用到的振荡器来说，非层次化的符号化分析仍然能够提供理想的计算效率。

除了系数计算以及求根过程，符号化分析方法的首要步骤是建立与电路拓扑结构相对应的数据结构。从表 5-4 中可以看出，建立数据结构的时间占据了整个符号化分析方法的大部分时间。然而，这个数据结构一旦建立后，当设计者需要调整电路中的参数以查看根轨迹的变化情况时，便可利用这一数据结构进行重复的分析，无需再重建数据结构。

如果我们尝试调整图 5-4 中两个偏置电感的大小，则可以用同一个符号化数据结构计算主极点根轨迹的分布状况。得到的结果如图 5-7a)所示。我们将偏置电感不同时所对应的相位噪声曲线放在图 5-7b)中作对比分析。

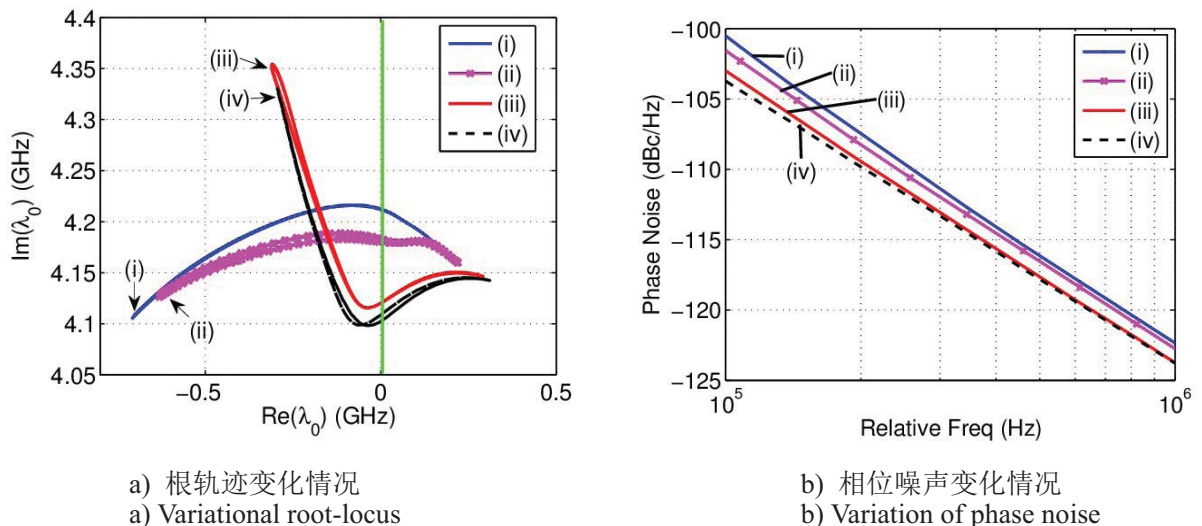


图 5-7 调整偏置电感对振荡器 2 的性能影响

Figure 5-7 The effect of biased inductors on the performance of oscillator 2

图中，轨迹(i)表示 $L_{top}=0$, $L_{bot}=0$ 的情况，轨迹(ii)表示 $L_{top}=0.8nH$, $L_{bot}=0$ 的情况，轨迹(iii)表示 $L_{top}=0$, $L_{bot}=1.8nH$ 的情况，轨迹(iv)表示 $L_{top}=0.8nH$, $L_{bot}=1.8nH$ 的情况。可以看到，在两个偏置电感的作用下，根轨迹逐步向虚轴靠拢，两个偏置电感极大地限制了根轨迹深入左半平面的程度，因此相位噪声性能也逐步变好。

用符号化仿真器做这类拓扑结构固定，调节电路元素的分析，相对于数值方法更为方便。

5.2 主极点根轨迹敏感度分析结果

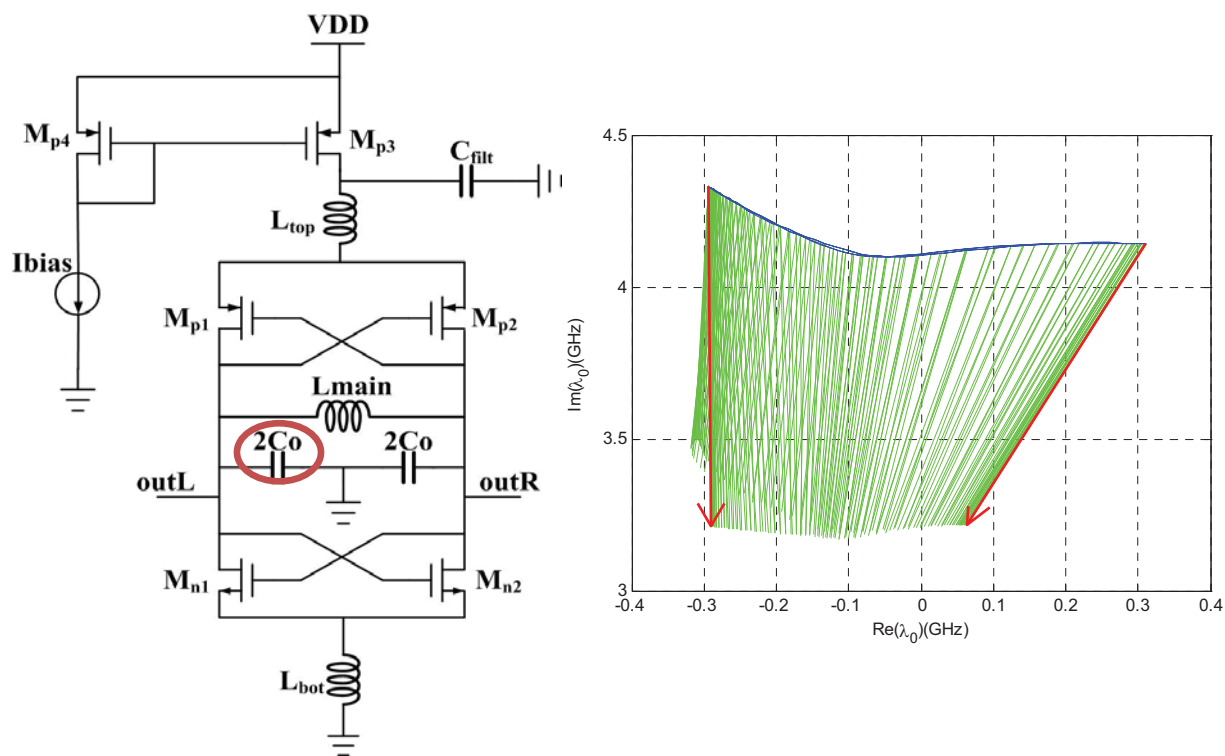


图 5-8 根轨迹变化对 C_o 的敏感度
Figure 5-8 Sensitivity of root locus in reference to C_o

我们依然以图 5-2 中的带电感偏置的振荡器作为例子。如图 5-8 所示，我们可以看出， C_o 对根轨迹变化的影响主要在于根轨迹的虚数部分。增大 C_o ，根轨迹将会整体沿着虚轴方向向下移动。这反映了振荡器振荡频率的降低。由于振荡器的频率大约为 $\frac{1}{\sqrt{L_{main}C_o}}$ ，因此，这一敏感度信息与理论相符。

同时我们可以从红色的敏感度信息中看到，增大 C_o ，会使轨迹向虚轴靠拢。从理论分为可知，当理想的 C_o 变大时，振荡器的 Q 值将会增大，从而能够带来更优的功耗及相噪性能。这一点亦与理论相符。

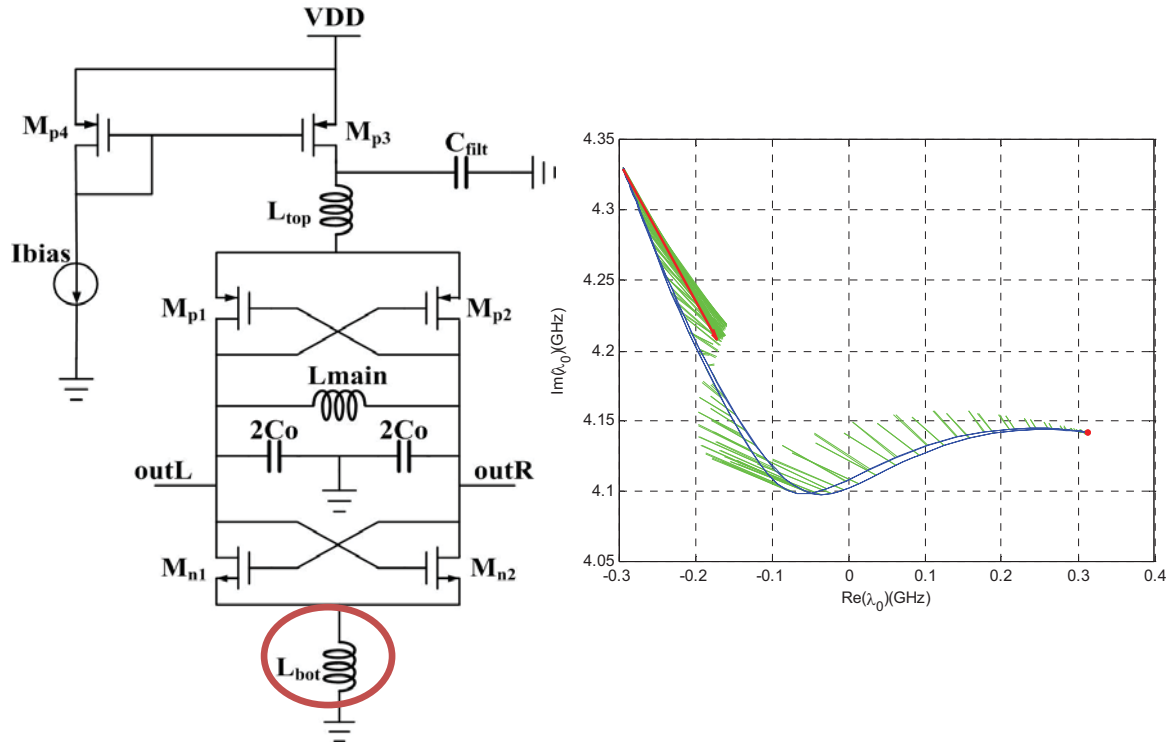


图 5-9 根轨迹变化对 L_{bot} 的敏感度
Figure 5-9 Sensitivity of root locus in reference to L_{bot}

如图 5-9 所示，我们可以看出， L_{bot} 可以使主极点向虚轴靠近。 L_{bot} 对根轨迹的影响主要表现在左半平面上，对右半平面的影响相对较小。当 L_{bot} 增大时，能够拉动左半平面中，根轨迹的极左值点向虚轴靠近，从而限制根轨迹在左半平面的深入程度，以减少相位噪声。这一预测亦与理论相符。

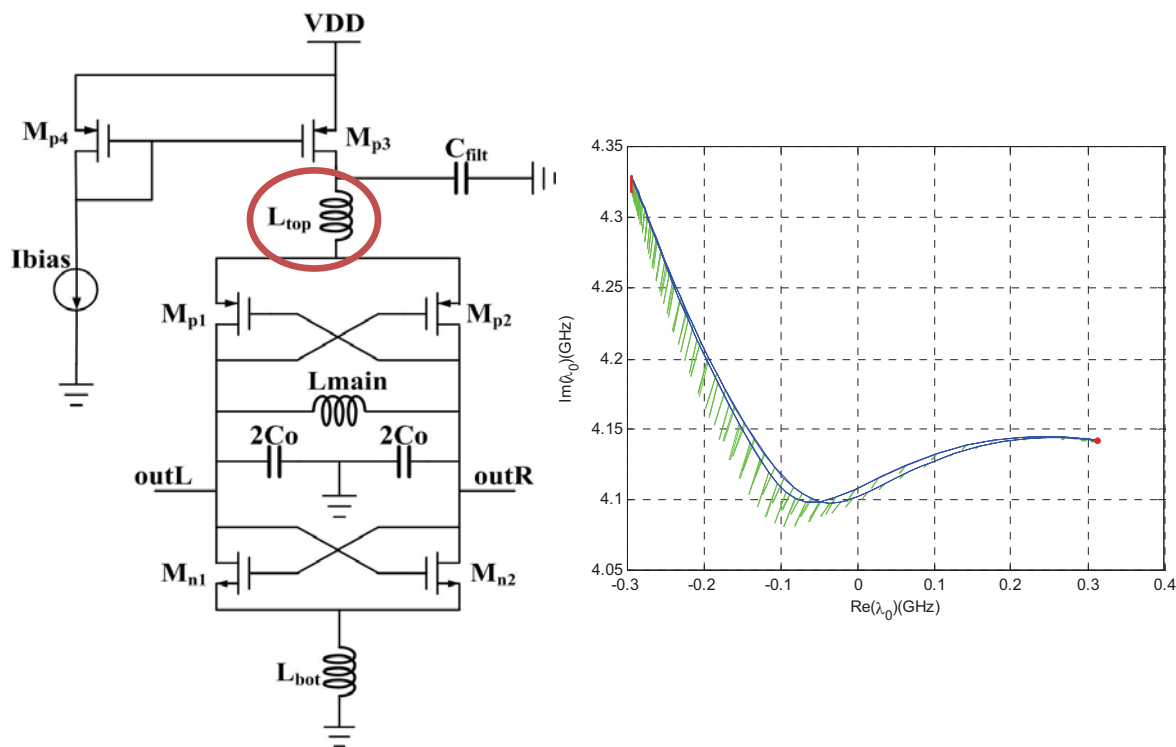


图 5-10 根轨迹变化对 L_{top} 的敏感度
Figure 5-10 Sensitivity of root locus in reference to L_{top}

如图 5-10 所示，我们可以看出， L_{bot} 对根轨迹的影响主要亦表现在左半平面上。当 L_{bot} 增大时，能够拉动左半平面的极左值点向虚轴靠近（红色的极值点实数部分为正），从而限制根轨迹在左半平面的深入程度，以减少相位噪声。这一预测亦与理论相符。

我们选取根轨迹的左极值点，按敏感度的幅度 $Mag(Sens(s_{leftmost}, p))$ 的大小对电路中的参数值 p 进行排序，可以得到如表 5-5 所示的结果。

表 5-5 对根轨迹左极值点最敏感的电路元件

Table 5-5 Circuit Parameters that are most sensitive to the leftmost point of the root locus

电路元件	C_0	L_1_Lmain	L_0_Lmain	C_1
敏感度幅值 ($2\pi * 10^9$)	1.17653	0.991	0.989	0.838
电路元件	L_{bot}	Rsd_M_{n2}	R_1_Lmain	R_0_Lmain
参数值	0.173	0.103	0.102	0.101
电路元件	Gm_M_{p1}	Cgg_M_{n2}	Rsd_M_{p1}	Gm_M_{n2}
参数值	0.087	0.080	0.060	0.052
电路元件	Cgg_M_{p1}	L_3_Lmain	L_2_Lmain	R_3_Lmain
参数值	0.048	0.030	0.030	0.025
电路元件	R_2_Lmain	C_2_Lmain	Cgg_M_{p2}	Cdd_M_{n1}
参数值	0.025	0.021	0.015	0.015
电路元件	Cgd_M_{n2}	L_{top}	Cdg_M_{n2}	Cgd_M_{p1}
参数值	0.012	0.012	0.009	0.008

我们将这一结果表示在图 5-11 中。

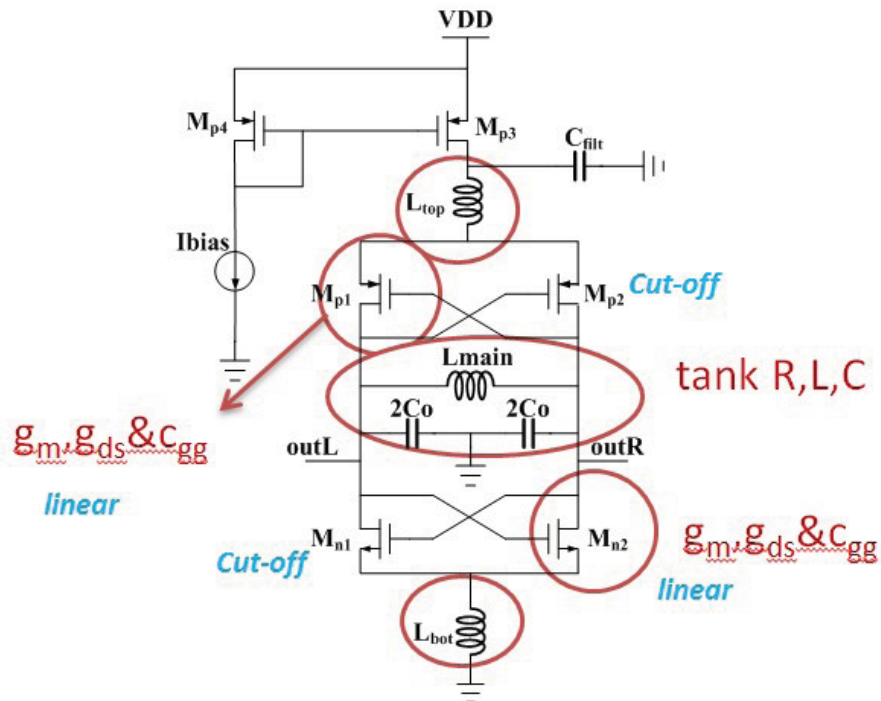


图 5-11 对左极值点影响最大的参数示意图

Figure 5-11 Parameters that impact the left-most point in the main root-locus the most

我们发现，根轨迹左极值点对两个偏置电感、振荡器的 LC tank 以及处于线性区的两个 MOS 管的内部小信号参数均比较敏感。

我们再选取根轨迹的右极值点，按敏感度的幅度 $Mag(Sens(s_{rightmost}, p))$ 的大小对电路中的参数值 p 进行排序，并挑选出敏感度最大的数值，可以得到如表 5-6 所示的结果。

表 5-6 对根轨迹右极值点最敏感的电路元件

Table 5-6 Circuit Parameters that are most sensitive to the rightmost point of the root locu

电路元件	L_{1_Lmain}	L_{0_Lmain}	C_0	C_1
敏感度幅值 ($2\pi * 10^9$)	1.038	1.037	0.906	0.906
电路元件	$Gm_{M_{n2}}$	$Gm_{M_{n2}}$	$Gm_{M_{n2}}$	$Gm_{M_{n2}}$
参数值	0.190	0.190	0.165	0.162
电路元件	R_{1_Lmain}	R_{0_Lmain}	$C_{gg_M_{p2}}$	$C_{gg_M_{p1}}$
参数值	0.110	0.110	0.035	0.034
电路元件	L_{3_Lmain}	L_{2_Lmain}	R_{3_Lmain}	R_{2_Lmain}
参数值	0.028	0.028	0.025	0.025
电路元件	$C_{gg_M_{n1}}$	$C_{gg_M_{n2}}$	C_{2_Lmain}	
参数值	0.024	0.024	0.021	

我们将这一结果表示在图 5-12 中。

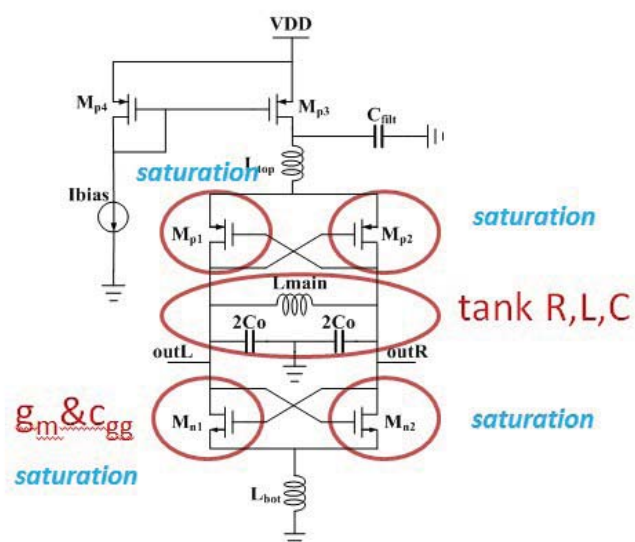


图 5-12 对右极值点影响最大的参数示意图

Figure 5-12 Parameters that impact the right-most point in the main root-locus the most

我们发现，根轨迹右极值点对四个处于饱和状态 MOS 管的 g_m 、 C_{gg} ，振荡器的 LC tank 比较敏感，对两个偏置电感则不敏感。对比图 5-11 与图 5-12，我们可以得出结论，振荡器的两个偏置电感 L_{top} 与 L_{bot} 主要影响根轨迹位于左半平面的部分，而对根轨迹在右半平面的分布相对较少。两个电感的主要作用是阻止根轨迹深入左半平面，从而降低电路中较大的正导纳所引入的噪声，使得相位噪声性能变好。这一分析指出了 L_{top} 与 L_{bot} 是电路性能优化的关键因素，需要调节这两个元件的值，使电路的相位噪声达到最佳性能。

需要指出的是，由于没有考虑到大信号变化的影响，利用符号化分析方法分析振荡器主极点根轨迹相对电路元件的敏感度时，需要保证在细微调节电路中的元件时，振荡器的 PSS 结果不会发生太大变化。否则，利用这一结果对电路进行优化分析，有可能导致一定的误差。

5.3 符号化综合方法效率分析

在前述的实验过程中，我们利用符号化分析方法分析同一个振荡器电路，能够有效地给出分析结果，提供电路优化信息。在这一部分，我们主要分析符号化综合方法添加及删除原件的效率，并将这一效率与未使用综合综合方法的符号化数据结构构建过程进行比较。

假设我们需要对 Osc-1 和 Osc-2 这两种不同的结构进行比较。

首先给出通过元件极限化操作从建立好的 Osc-2 中计算 Osc-1 电路特征多项式系数的效率。两个偏置电感短路,值设为 0、电容 C_{fit} 开路，值也置为 0。

表 5-7 基于两种方法的 Osc-1 系数计算时间比较
Table 5-7 Comparison of Osc-1 coefficient calculation time based on two methods

	基于 Osc-1 重构	基于 Osc-2 的元件极限化操作
Osc-1 符号化数据结构建立时间(s)	0.2	0
Osc-1 系数计算时间(s)	0.058	0.067
总时间(s)	0.258	0.067

基于 Osc-2 元件极限化操作进行 Osc-1 的系数计算时，由于判断条件增多，并且需要访问的 GPDD 节点数量有所增加，因此效率比起基于 Osc-1 重构的系数计算有所降低。但是由于极限化操作使得 Osc-1 避免了重新构造 GPDD 及 s 展开

BDD 的时间，因此总体的时间效率大大提升了。

我们再来分析在 Osc-1 基础上添加两个电感和一个电容生成 Osc-2 的效率。我们将被添加的电感和电容的顺序排在其他电路元件符号之前，每次添加一个元件生成新的 GPDD 后，我们均做一次 Reduce 和 Zero Suppress 操作。得到的结果如表 5-8 所示。

表 5-8 元件添加操作细节统计
Table 5-8 Details of the adding devices in symbolic method

	原始 GPDD 大小	增加 C_{filt} 所增加的 GPDD 大小	增加 L_{bot} 所增加的 GPDD 大小	增加 L_{top} 所增加的 GPDD 大小	总 GPDD 大小	生成总时间(s)
从 Osc-1 变化到 Osc-2	642	298	1216	1368	3524	0.34

通过这种元件添加的方法生成的 GPDD 结构可用于分析 Osc-1 及 Osc-2 两个电路。而使用传统方法时，我们需要分别构造 Osc-1 和 Osc-2 两个电路，构造时间分别为 0.08s 和 0.33s，共 0.42s。使用元件添加的方法进行电路的对比分析，多个电路的总构造时间可缩减为其中最大的电路的构造时间，效率大大提升。

从以上分析中可以看出，用集成电路综合方法对电路做对比分析，能够大大缩短仿真时间，提高分析效率。

5.4 本章小结

本章主要测试了两个不同的振荡器结果，分析了符号化主极点根轨迹分析方法的效率。

根据符号化方法提供的敏感度信息，本章指出了影响根轨迹分布的关键因素，从而为电路设计提供指导信息。

本章中亦测试了符号化综合方法在电路拓扑结构修改中的作用，对符号化综合方法的效率作了较为详细的分析。

第六章 结束语

6.1 主要工作与创新点

本文提出了一套符号化振荡器主极点根轨迹的分析方法，并在此基础上提出了根轨迹敏感度的计算方法。

这一方法能够利用振荡器在周期性小信号分析中拓扑结构不变的特性，加速主极点根轨迹的求解过程，并为振荡器的优化指出提升性能的关键因素。

为了方便比较不同电路的性能，本文还提出了一套符号化电路综合方法，使得符号化结构能够根据电路中元件的添加与删除进行灵活的变化。这一套综合方法不仅适用于振荡器电路的比较分析，还能够适用于各类模拟、射频电路的符号化分析，能够有效地比较近似电路结构的性能。

本文的工作，是对实验室之前的研究所作的拓展以及进一步发展。应用本文提出的方法，能够对不同的高频振荡器结构进行直观的性能分析。更进一步的是，本文提出了一套全新的方法，能够根据电路的拓扑结构变化，对符号化数据结构进行动态的调整，进一步提升了符号化数据结构的构造效率，为进一步的研究和发展建立了坚实的基础。

6.2 后续研究工作

本文初步探索了符号化方法在高频振荡器仿真领域的作用，并提出了高频振荡器主极点根轨迹的敏感度分析方法。符号化方法在这一类的分析中有独特的优点，能够解释不同的电路元件对于振荡器主极点根轨迹变化所起到的作用，从而进一步为设计者提供有用的指导信息。但是，这一敏感度求解方法是建立在改变元件值时，电路中其他小信号参数值不会受到太大的基础上进行计算的。当所调节的电路元件对振荡器 PSS 状态起到较大影响时，用本文所述的方法求解敏感度信息，会有一定的误差。

另外，本文的大信号分析依赖于 Cadence 仿真工具，而大信号分析受到了直流工作点分析速度的制约，效率不高。在进一步的研究中，我们希望能够建立一套更完备的 PSS 分析工具，与符号化分析工具进行较完美的结合，能够快速提供 MOS 管小信号参数值，计算 PSS 的敏感度信息，与本文所提供的计算方法结合起来，形成一套自动化程度更高的仿真工具，为振荡器电路的设计带来更多的便利。

我们尝试通过调节电路中的元件值，根据根轨迹的变化情况来优化在固定拓扑结构下的功耗及噪声性能。但是由于目前时变根轨迹的文献不多，分析只能定性地进行，定量的对应关系至今仍不清楚。通过本文的敏感度计算和比较，我们能够清晰地看出振荡器所走过的轨迹在不同时刻点受哪些元素影响比较大，并且可以看出轨迹变化的方向。但是我们无法通过根轨迹得到电路最优化的结果。优化电路仍然需要结合仿真器的相噪分析等操作来进行。我们希望能够在后续的研究过程中，进一步清晰化时变根轨迹与电路性能之间的确切联系，以进一步为电路分析带来一些定量的信息。

最后，本文只为符号化综合方法介绍了一个简单应用。实际上，这一套综合方法能够使用于许多值得研究的课题。我们希望在不久的将来，能够提供一套便于用户使用的电路网表编辑工具，与本文中提到的综合方法结合起来，使得在用于对电路进行元件添加、删除等操作时，仿真器能够快速建立新的拓扑结构，提高仿真的效率。我们也希望在这一综合方法的基础上，符号化仿真工具能够在更大程度上发挥其优势，为模拟电路的设计与自动化提供更多便利。

参 考 文 献

- [1] L. Zadeh, "Frequency analysis of variable networks," in *Proc. IRE.*, vol. 32, pp. 291–299, 1950.
- [2] O. Nastov, R. Telichevesky, K. Kundert and J. White, "Fundamentals of fast simulation algorithms for RF circuits," *Proc. IEEE*, vol. 95, pp. 600–621, 2007.
- [3] K. Mayaram, D. C. Lee, S. Moinian, D. A. Rich, and J. Roychowdhury, "Computer-aided circuit analysis tools for RFIC simulation: Algorithms, features, and limitations," *IEEE Trans. Circ. Syst. II*, vol. 47, pp.274–286, 2000.
- [4] W. R. Evans, "Control system synthesis by root locus method," *Amer Institute Electrical Engineers (AIEE) Trans*, Part II, vol. 69, pp.14, 1950.
- [5] N. M. Nguyen and R. G. Meyer, "Start-up and frequency stability in high-frequency oscillators," *IEEE J. Solid-State Circuits*, vol. 27, no. 5, pp. 810–820, May 1992.
- [6] A. Bevilacqua, F. Pavan, C. Sandner, A. Gerosa, and A. Neviani, "Transformer-based dual-mode voltage-controlled oscillators," *IEEE Trans. Circuits Syst. II*, vol. 54, no. 4, pp. 293–297, Apr 2007.
- [7] A. Buonomo and A. Lo Schiavo, "Analysis of emitter (source)-coupled multivibrators," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 6, pp. 1193–1202, Jun. 2006.
- [8] T. Okamoto, Y. Maruyama, and A. Yukawa, "A stable high-order deltasigma modulator with an FIR spectrum distributor," *IEEE J. Solid-State Circuits*, vol. 28, no. 7, pp. 730–735, Jul. 1993.
- [9] S. S. Broussev and N. T. Tchamov, "Time-varying root-locus of large-signal LC oscillators," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 5, pp. 830–834, May 2010.
- [10] J. Vlach and K. Singhal, "Network functions in the frequency domain," in *Computer Methods for Circuit Analysis and Design*, pp. 267–288, 2003.
- [11] S. Hayley, "The generalized eigenproblem: Pole-zero computation," in *Proc. IEEE.*, vol. 76, no. 2, pp. 103–120, Feb 1988.
- [12] G. Gielen, P. Wambacq and W. Sansen, "Symbolic analysis methods and applications for analog circuits: A tutorial overview," in *Proc. IEEE*, vol. 82, pp. 287–303, Feb 1994.
- [13] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Trans. Comput.*, vol. C-37, pp. 677–691, Aug 1986.
- [14] K. S. Brace, R. L. Rudell and R. E. Bryant, "Efficient implementation of a BDD

- package”, *Proceedings of the 27th ACM/IEEE conference on Design automation*, p.40-45, June 24-27, 1990, Orlando, Florida, United States
- [15] C.-J. Shi and X.-D. Tan, “Canonical symbolic analysis of large analog circuits with determinant decision diagrams,” *IEEE Trans. on Computer-Aided Design*, vol. 19, no. 1, pp. 1-18, Jan 2000.
- [16] G. Shi, W. Chen and C.-J. Shi, “A Graph Reduction Approach to Symbolic Circuit Analysis,” in *Proc. Asia and South-Pacific Design Automation Conference (ASPDAC)*, Yokohama, Japan, pp. 197-202, Jan 2007.
- [17] W. Chen and G. Shi, “Implementation of a Symbolic Circuit Simulator for Topological Network Analysis,” in *Proc. IEEE Asia Pacific Conference on Circuit and System (APCCAS)*, Singapore, pp.1327-1331, Dec 2006.
- [18] 陈微微, “符号化模拟电路仿真器的实现与应用,” 上海交通大学微电子学院硕士学位论文, 2007年3月。
- [19] D. E. Muller, “A method for solving algebraic equations using an automated computer,” in *Mathematical Tables and Other Aids to Computation*, vol. 10, pp. 208–215, 1956.
- [20] T. H. Lee, "The Design of CMOS Radio-Frequency Integrated Circuits", Cambridge University Press, Dec 22, 2003.
- [21] B. Broussev, "High-Performance LC-VCOs and their Analysis using Time-Varying Root-Locus", Ph. D. Dissertation, May 2012, URN: <http://URN.fi/URN:ISBN:978-952-15-2865-1>
- [22] J. Vlach and K. Singhal, "Network functions in the frequency domain", in *Computer Methods for Circuit Analysis and Design*, 2nd ed. Norwell, MA: Kluwer, 2003, ch. 7, pp. 267-288.
- [23] G. Anderson, and P. Lin, "Computer generation of symbolic network functions – A new theory and implementation", *IEEE Trans. Circuit Theory*, vol. 20, no. 1, pp. 48-56, Jan 1973.
- [24] B. Bollig and I. Wegener, "Improving the Variable Ordering of OBDDs Is NP-Complete, " *IEEE Transactions on Computers*, v.45 n.9, p.993-1002, September 1996.
- [25] *HSPICE User Guide: RF Analysis*, Synopsys, Inc., Mountain View, CA, December 2010, version E-2010.12.
- [26] *Virtuoso Spectre Circuit Simulator Reference*, Cadence Design Systems, Inc., San Jose, CA, December 2009, product version 7.2.
- [27] K. Mayaram, D. C. Lee, S. Moinian, D. A. Rich, and J. Roychowdhury, “Computer-aided circuit analysis tools for RFIC simulation: algorithms, features, and

- limitations,” *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 4, pp. 274–286, Apr. 2000.
- [28] Y. Tsividis, *Operation and Modeling of the MOS Transistor*, 2nd ed. Oxford, UK: Oxford University Press, 1999.
- [29] G. Shi and X. Meng , "Variational analog integrated circuit design via symbolic sensitivity analysis," *IEEE International Symposium on Circuits and Systems*, pp.3002-3005, 24-27 May 2009.
- [30] J. Chen, G. Shi, A. Tai and F. Lee, “A size sensitivity method for interactive MOS circuit sizing,” *IEEE 10th International New Circuits and Systems Conference (NEWCAS)*, pp.169-172, June 2012.
- [31] X. Liu, S. Tan, and H. Wang, “Parallel statistical analysis of analog circuits by GPU-accelerated graph-based approach,” in *Proc. Design, Automation and Test in Europe (DATE)*, Dresden, Germany, 2012, pp. 852–857.
- [32] H. Xu, G. Shi, and X. Li, “Hierarchical exact symbolic analysis of large analog integrated circuits by symbolic stamps,” in *Proc. ACM/IEEE Asia South-Pacific Design Automation Conference (ASP-DAC)*, Yokohama, Japan, Jan. 2011, pp. 19-24.

致 谢

转眼，在交大已经度过六个春秋。在这人生最美好的时光中，我很庆幸能够认识到每一位指导过我的老师们，以及与我共享喜怒哀乐的同学朋友们。是你们，让我学会了严谨、认真、执著的生活态度，陪我度过了精彩的校园生活，让我逐渐成长。

在这一段成长过程中，给我影响最深的是我的导师施国勇教授。从大三的 DSP 课上第一次接触到他，我就感受到了一种威严，一种气场。后来我才发现，这种气场是来自于一种执着认真的治学态度，来自于几十年如一日的坚持。每天，他总是早早地来到办公室，查邮件，读论文，写代码，做 PPT，写论文，做好每一步，享受研究所带来的喜悦。每当他有新的想法时，他总是乐于亲自跑到实验室来与我们分享，帮助我们拓展研究思路。每当我遇到难题时，他总是能用简单明了的数学方法给我解答，为我带来新的启发。我很荣幸能成为他的学生，在他的指导下，一步一步地走过从课题的提出，到算法的设计，到有效地实现代码，到发表论文的整个过程。对我来说，受益终身的不只是一些专业知识，一种思维方式，更是一份坚持，一份无论浮华的世界如何变化，都踏踏实实，勤勤恳恳地坚持着自己事业的态度。施老师，谢谢您。

同时要感谢的还有李章全老师。他同样有着一份异于常人的坚持，无论做学术，还是做人。从他身上，我看到了一种对知识和对生活的热情，看到了一种永不妥协的精神。谢谢您，让我认识到了什么是一个真正的工程师。

我也要感谢祝永新老师，感谢您在课堂上的谆谆教导以及在课下给我耐心指导和鼓励。感谢黄田，感谢冬阳，感谢倩楠，感谢严顺卿。和你们一起讨论研究、一起做项目的时光，让我终生难忘。

感谢 EDA 实验室的师兄师姐和师弟们。感谢郝志刚学长、马迪铭学长、黄伟坚学长和李小鹏学长，从你们身上我看到了一种对自己理想的执著。感谢徐辉，从你身上我看到了智慧，看到了一种认真细致的工作态度，看到了活力和热情。感谢张贺学姐、彬彬姐和爱林姐，能够和你们一起学习、一起成长，是我的荣幸。感谢工作效率超高，代码写得飞快的宋阳，感谢热爱天文热爱足球也热爱编程的陈家俊，感谢一丝不苟的兰兰姐，很庆幸能够和你们和你们一同做研究，祝你们拥有似锦的前程。感谢程建东，你的认真和努力让我感动。感谢胡翰彬和陈静，你们是实验室新的希望。祝爱林姐、建东、陈静及翰彬将研究越做越好，将 EDA

实验室的精神发扬光大。祝大家都能实现自己的理想，拥有灿烂的事业。

感谢微电子学院的老师们和同学们。在这里，我遇到了一群好老师，收获了一生的挚友。祝愿你们都能够拥有幸福的生活。

最后，我要感谢我生命中最重要的两个人。爸爸妈妈，谢谢你们对我的爱以及无私的付出。在任性的女儿为了兴趣和理想远赴上海求学时，你们即使心中不舍，依然无怨无悔地支持我。无论我遇到什么挫折，你们总是能够为我提供最温暖的港湾。我爱你们。你们的支持，永远是我前进的动力。

攻读硕士学位期间已发表或录用的论文

- [1] Y. Zhu, G. Shi, A. Tai and F. Lee, "Symbolic Time-Varying Root-Locus Analysis for Oscillator Design," *IEEE 10th International New Circuits and Systems Conference (NEWCAS)*, pp.165-168, June 2012.