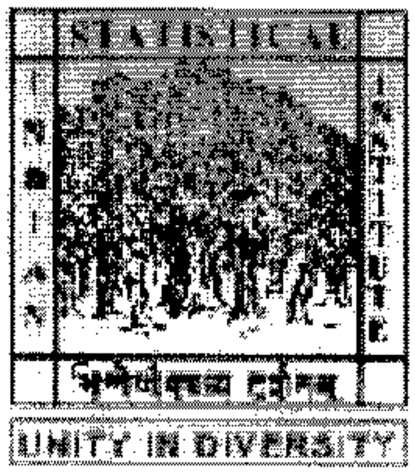
M.Tech. (Computer Science) Dissertation Series

POWER GRID SIMULATION

a dissertation submitted in partial fulfillment of the
requirement for the M.Tech. (Computer Science)
degree of the Indian Statistical Institute

By
I.Vishnu Vardhan

under the supervision of
Dr. Susmita Sur-Kolay
Advanced Computing and Microelectronics Unit



INDIAN STATISTICAL INSTITUTE
203, Barrackpore Truck Road
Kolkata 700 108
July 2005

Certificate of Approval

This is to certify that this thesis titled "Power Grid Analysis" submitted by I.Vishnu Vardhan towards partial fulfillment of requirements for the degree of M.Tech in Computer Science at Indian Statistical Institute embodies the work done under my supervision.

The dissertation report may be accepted in partial fulfillment of the requirement for the M.Tech (Computer Science) degree of Indian Statistical Institute, Kolkata, India.

(Supervisor)

(External Examiner)

Acknowledgement

I would like to acknowledge and express my gratitude to all those who have supported me directly or indirectly, to complete this dissertation work.

First, I would like to thank my guide Dr. Susmita Sur-Kolay. It was a pleasure working with her. Her suggestions have been invaluable for completing this work.

I also wish to thank the support of Sandeep, Debasis, Suchandra Roy, Subhra Lahiri, Sambhu, Subhasis, Pritha and many others, who are Project Linked Personnel and Research Scholars at ACMU, ISI, for their encouragement and assistance.

Special thanks to my classmates who helped me in many ways, especially Deepak Kumar.

Kolkata, July 2005

I. Vishnu Vardhan

Contents

1. INTRODUCTION

VLSI system performance has increased by orders of magnitude in the last few decades. Continued technology scaling and improving transistor performance to increase frequency has made it possible. Increasing integration capacity enabled designers to realize complex architectures, and reducing energy consumed per logic operation to keep power dissipation within limit. The technology treadmill will continue and soon provide integration capacity of billions of transistors. The technology trend of integrating number of transistors was predicted by Moore and is presented below in Figure 1.1. However, this increase in number of devices has made power consumption as barrier. So, this has made the power delivery networks in VLSI circuit design a major design challenge.

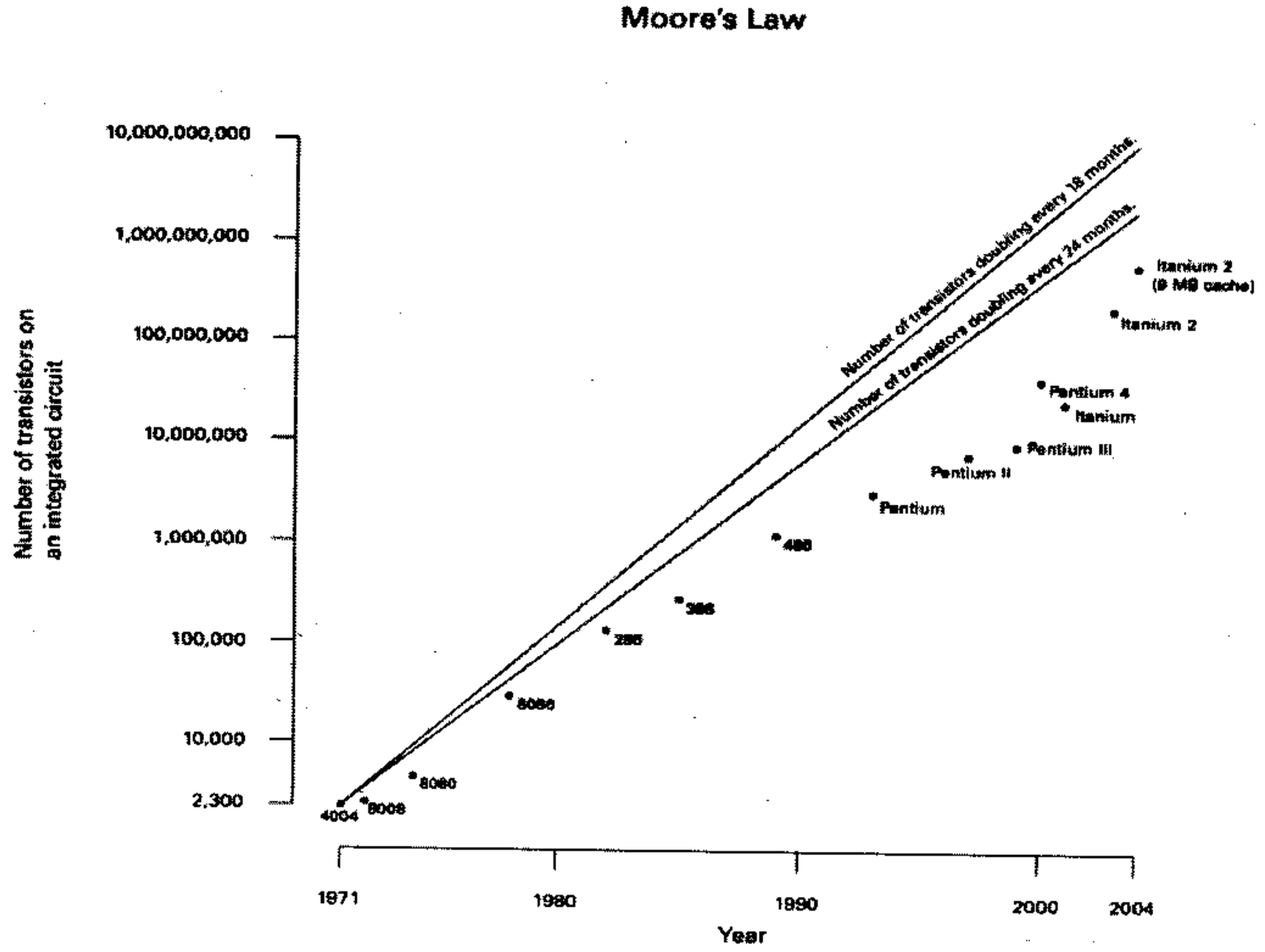


Figure 1.1: Moore's Law

In micron technology, chips used to have two metal layers for routing signals as well as power and ground. This called for special power and ground routing algorithms and these were usually run before signal routing. But in nanometer

technology as the number of devices on chip is increasing, we need to have not only multiple metal layers for proper routing of these nets to devices but also a regular grid like layout for the power delivery network on almost all the metal layers. The power grid is composed of alternate metal lines of power (Vdd) and ground (GND) nets in each layer. The upper and lower metal layers are connected by vertical vias. Typical power grid with three layers is shown in Figure 1.2. Vdd and GND terminals of transistors connect to the lower metal layer vias.

The currents and voltages can be analyzed at the via's. Simultaneous switching of transistors connected to the via causes maximum current flow through the via and a droop in voltage. Voltage droop at the via slows the switching of transistors connected to it. If the voltage droop at the via is more than stimulated value, then it makes the circuit faulty and is elaborated below.

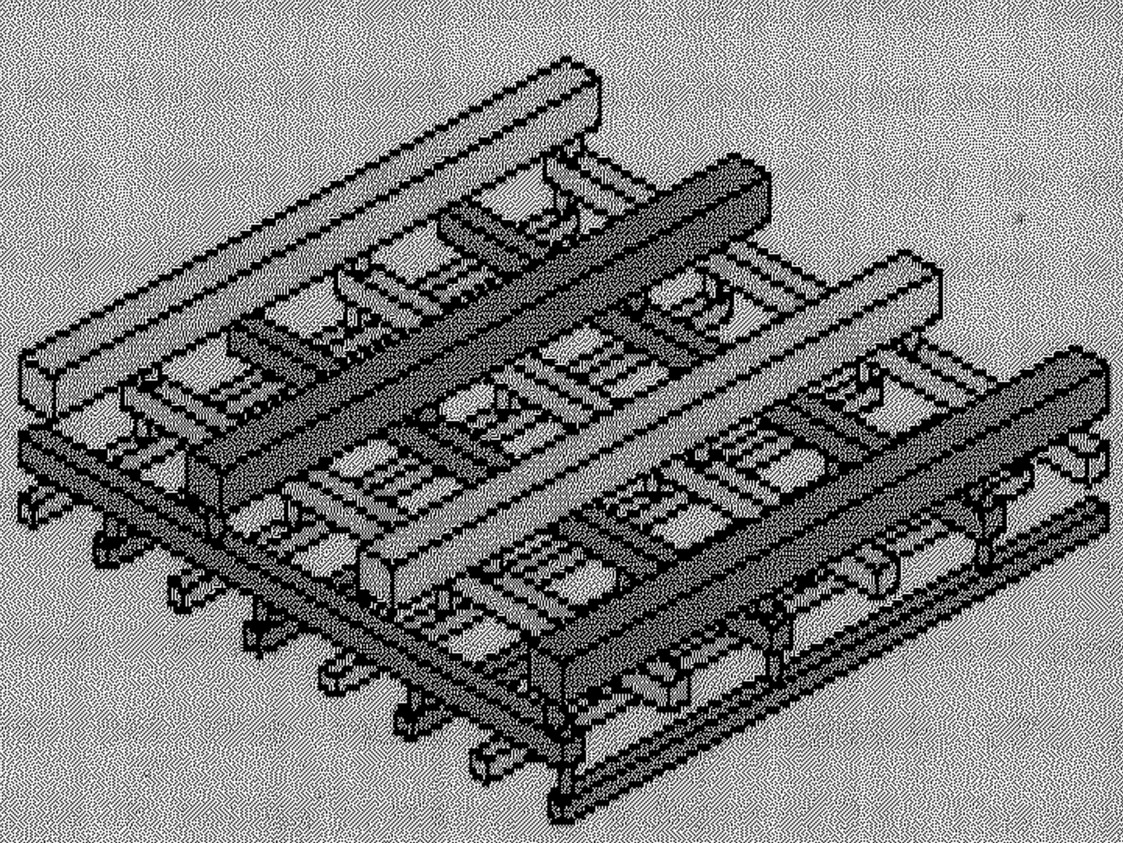


Figure 1.2: Typical Power grid

To lower the total power dissipation, there is also reduction in supply voltage and the corresponding reduction in device threshold voltage. Thus are circuits more susceptible to noise, including power bus voltage variations. Also, increased density of switching devices and rising frequency has lead to power density problem. The rise in power density with a simultaneous reduction in power supply voltage leads to a large increase in the amount of current that needs to be delivered.

The current demand is increasing as specified above but the supply voltage of modern sub micron VLSI is decreasing day by day. This results in huge power grids that are required to distribute large amounts of current, at increasingly lower voltages. The resulting voltage drop on the grid reduces noise margin and increases gate delay, having a serious performance impact. Reduced noise margins may lead to false switching at certain logic gates and latches. Higher logic gate delays, on the other hand, may slow down the circuit enough so that timing requirements cannot be met. Hence, once voltage drop exceeds certain designer-specified threshold like 30 mv, there is a high probability of faulty behavior of the circuit.

Some sources of power fluctuation are *IR* drop, *Ldi/dt*-drop, and resonance issues. Traditionally power grid analysis is often emphasized on IR-drop. Recently, due to the rapidly increasing operation frequency, the dynamic power fluctuation caused by *LdI/dt* has also become significant. Large numbers of on-chip decoupling capacitors are added to act as temporary on-chip local power supplies for reducing the power fluctuation. Hence, to accurately model and verify the quality of power delivery, both capacitance and inductance should be considered in the power grid analysis problem. With increasing clock frequency, the time period and the time margins are reduced. Therefore, any uncertainty in the gate delay has to be modeled and accounted for in the design process.

It has been observed by many researchers that the power density of high–end microprocessors has been increasing by approximately 80% per technology generation, while the voltage is scaling by a factor of 0.8. This is leading to 225% increase in current per unit area in successive generation of technologies. Therefore as the current is increasing, the IR drop is also increasing.

At present modern VLSI circuit power grids are becoming performance-limiting factors. Hence, efficient analysis of power grids is necessary for both predicting the performance and improving the performance if possible. For analysis of power grids traditional circuit simulation techniques are not practical with respect to time and memory complexity. Hence, new efficient techniques for power grid

analysis are to be developed keeping into consideration the execution time and memory.

1.1 MOTIVATION OF THIS WORK

The degree of scaling in VLSI technology and ever increasing integration density have led to unprecedented levels of power dissipation in current day VLSI circuits, and possibly creating a barrier to further scaling and integration unless effective design techniques are developed to reduce power. So, tools for power grid analysis are required to designers in order to analyze this problem.

SPICE is a free software tool used for power grid analysis. But we cannot use SPICE, as it cannot handle circuits with large number of nodes. So we need to have tools with efficient techniques for power grid analysis satisfying following properties:

- Ability to handle millions of nodes
- Methods for extraction of the R L C values for the distributed system should be able to integrate into this simulation tool.
- Speed as well as accuracy is required in the simulation
- Multi-tiered speed-up
    - Hierarchical reduction of matrix
    - Cluster computing environment

The theme of this dissertation is to have new efficient techniques for power grid analysis considering the reduction in supply voltage, increasing frequency and increase in integration of devices and having the above properties. So, it helps the designers for the proper modeling of the power delivery network.

2. TECHNIQUES FOR FORMULATING EQUATIONS

We have already mentioned that simultaneous switching of transistors connected to via causes maximum current flow through the via and a droop in voltage. Voltage droop at the via slows the switching of transistors connected to it. Hence, via's are important for analysis of branch current and voltage droop. Specifically, via's at the lower metal layers are important for analysis as transistors are directly connected to them.

Extraction tools like Fastcap and Fasthenry are used to derive an RLC model of the power grid for circuit simulation. In order to better understand the effect of simultaneous switching, power grid analysis tool is required, for which, we need to formulate the circuit equations.

The method by which circuit equations are formulated is of key importance to a computer-aided circuit analysis and design program for integrated circuits. It affects significantly the set-up time, the programming effort, the storage requirements, and the execution speed of the computer program. The method which one selects needs to be flexible, computationally efficient, and economical with storage. Below we discuss some of the techniques for formulating the circuit equations.

2.1 Modified Nodal Analysis

Traditional nodal approach for formulating circuit equations treats voltage sources inefficiently and is incapable of including current-dependent elements, linear or nonlinear. Another disadvantage of traditional method is that branch currents are not accurately or conveniently obtained as part of the output of the program. Modified Nodal Analysis (MNA) is the most efficient way to formulate equations for circuits containing different types of network elements.

In formulating the circuit equations by using the MNA for a given network, we start with the set of branch currents through the voltage sources and through elements whose currents are controlling variables, and nodal voltages versus a common datum node as variables. Kirchhoff's current law is applied to each node other than the datum node in the circuit such that the summation of currents leaving the node is equal to zero.

The MNA matrix can in general be expressed in the form

$$\begin{bmatrix} Y_R & B \\ C & D \end{bmatrix} \begin{bmatrix} V \\ I \end{bmatrix} = \begin{bmatrix} J \\ F \end{bmatrix} \quad \text{--------} (1)$$

where Y_R is a reduced form of the nodal matrix excluding the contributions due to voltage sources, current controlling elements. B contains partial derivatives of the Kirchhoff current equations with respect to the additional current variables and thus contains +/-1's for the elements whose branch relations are introduced. The branch constitutive relations, differentiated with respect to the unknown vector, are represented by the matrices C and D. The vectors J and F are excitations which include the initial values from previous time steps corresponding to capacitors and inductors.

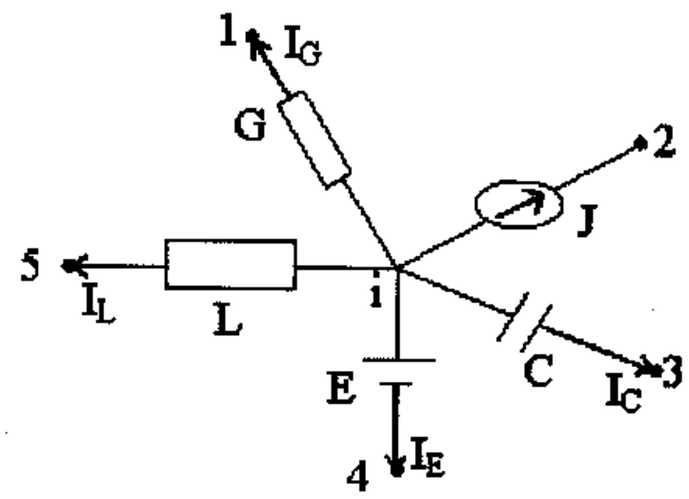


Figure 2.1: General Node

It is advantageous to consider the contributions of each circuit element to the MNA matrix separately. The "element rubber stamps" given in Table I corresponding to the circuit of a "general node" shown in Figure 2.1 summarize the contributions for each type of element, where BR refers to the additional branch relation and RHS is the contribution to the right side of (1).

The MNA matrix can be generated by using the element stamp table in a straightforward manner. For a given circuit, the matrix dimension is simply the sum of the number of nodes excluding the ground node plus the number of currents as outputs. We first arbitrarily label every node in some order then continue to label every element whose current is one of the outputs. We then collect all the elements in the network and process them one by one. For each element, depending on whether its current is an output or not, its contributions to the matrix can simply be read from the table and stamped into the matrix according to its node number or current number just labeled.

Capacitances and inductances are considered only in the time domain and their contributions, shown in Table I, are obtained by applying finite differencing methods to their branch relations. For example, the time derivative for the capacitor current $I_c(t) = C(dV_c/dt)$ is represented in terms of an implicit integration scheme as

$$I_c(t_n) = (C/h)(V_c(t_n)-V_{cp}) \text{ --------(2)}$$

where h is the time step and V_{cp} is a function of $V_c(t)$ at previous time points. Equation (2) can thus be represented by a conductance of C/h in parallel with a current source $C(V_{cp}/h)$. The element stamps for a capacitor C in Table I are thus found by using (2).

The branch current is always introduced as an additional variable for a voltage source, either independent or dependent, and an inductor and is thus readily available as an output variable. For current sources, resistors, and capacitors, this is only done under the following conditions:

- if other nonlinear circuit elements depend on its current; and
- if the branch current is requested as an output variable.

Stamp Table: Current not output Branch Current output

G		Vi	V1		RHS		Vi	V1	IG	RHS
	i	G	-G			i			1	
	1	-G	G			1			1	
						BR	G	-G	-1	
C		Vi	V3		RHS		Vi	V3	IC	RHS
	i	c/h	-c/h		c/h*vcp	i			1	
	3	-c/h	c/h		-c/h*vcp	3			-1	
						BR	c/h	-c/h	-1	C/h*vcp
L	L						Vi	V5	IL	RHS
						i			1	
						5			-1	
						BR	1	-1	-L/h	-L/h*ILp
E	E						Vi	V4	IE	RHS
						i			1	
						4			-1	
						BR	1	-1		E
J		Vi	V2		RHS		Vi	V2	IJ	RHS
	i				-J	i			1	
	2				J	2			-1	
						BR			1	J

Table I

The above method can be well explained by following example:

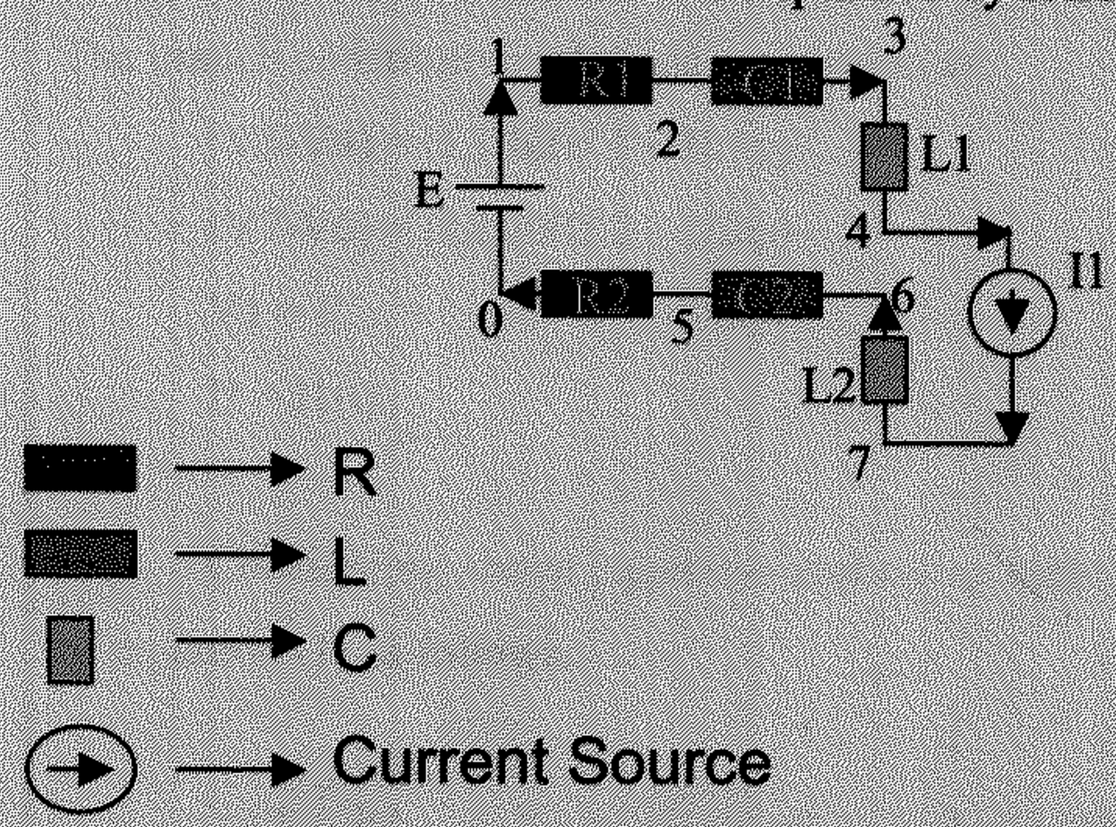


Figure 2.2: Example Circuit

Here R1 and R2 are the conductance of the resistors, C1 and C2 are the capacitance of the capacitors, L1 and L2 are the inductance values of the inductors, I1 is the current element following any waveform depending on time like I1=2*Sin(6.28*t), and E is the potential difference of the voltage source.

Let Vi and Vip be the present and previous time point nodal voltage at node i respectively. Let IL1, IL1p, IL2, IL2p be the present and previous time point currents through inductors L1 and L2 respectively, IE be the current through voltage source E and IJ be the current through the current source I1. Let us consider capacitors and resistors branch currents are not required. Let t be the present time point. Then, by applying Kirchhoff's current law and equation (2) to each node of above circuit, we get equations as follows:

R1(V1-V2) – IE = 0 ---- on applying to node 1

R1(V2-V1) + (C1/h)(V2-V3) = (C1/h)(V2p-V3p) ---- on applying to node 2

(C1/h)(V3-V2) + IL1 = (C1/h)(V3p-V2p) ---- on applying to node 3

-IL1 + IJ = 0 ---- on applying to node 4

R2(V5-0) + (C2/h)(V5-V6) = (C2/h)(V5p-V6p) ---- on applying to node 5

(C2/h)(V6-V5) – IL2 = (C2/h)(V6p-V5p) ---- on applying to node 6

IL2 – IJ = 0 ---- on applying to node 7

The branch constitutive relations are written as:

$$(V3-V4) = (L1/h)(IL1 - IL1p)$$

$$(V7-V6) = (L2/h)(IL2 - IL2p)$$

$$V1 = E$$

$$IJ = I1(t)$$

Considering above equations and stamp table I, we get the Y_R matrix on LHS of equation (1) as follows:

	1	2	3	4	5	6	7
1	R1	-R1	0	0	0	0	0
2	-R1	R1+C1/h	-C1/h	0	0	0	0
3	0	-C1/h	C1/h	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	R2+C2/h	-C2/h	0
6	0	0	0	0	-C2/h	C2/h	0
7	0	0	0	0	0	0	0

The above matrix does not contain contributions due to voltage source or any current controlling circuit element.

As matrix B on the LHS of equation (1) contains partial derivatives of the Kirchoff's current equations with respect to the additional current variables, we get the matrix B as:

	IL1	IL2	IE	IJ
1	0	0	-1	0
2	0	0	0	0
3	1	0	0	0
4	-1	0	0	1
5	0	0	0	0
6	0	-1	0	0
7	0	1	0	-1

As matrices C and D are constructed by differentiating branch constitutive relations with respect to unknown vector, we get matrix [C D] as:

	1	2	3	4	5	6	7	IL1	IL2	IE	IJ
IL1	0	0	1	-1	0	0	0	-L1/h	0	0	0
IL2	0	0	0	0	0	-1	1	0	-L2/h	0	0
IE	1	0	0	0	0	0	0	0	0	0	0
IJ	0	0	0	0	0	0	0	0	0	0	1

Vectors J and F on RHS of equation (1) contains the excitations, which include the initial values from previous time steps corresponding to capacitors and inductors. They are obtained from RHS of above Kirchhoff's equations as:

1	0
2	(C1/h)(V2p-V3p)
3	(C1/h)(V3p-V2p)
4	0
5	(C2/h)(V5p-V6p)
6	(C2/h)(V6p-V5p)
7	0
IL1	(-L1/h)IL1p
IL2	(-L2/h)IL2p
IE	E
IJ	I1(t)

Thus, all the above Kirchhoff's equations can be changed to matrices of equation 1 using the stamp table I. In this way, circuit equations of the given network can be formulated.

2.2 Hierarchical Analysis

The run time and memory requirement for solving a linear system is determined primarily by size and sparsity of the coefficient matrix. If the network is very large (10^7 - 10^8 nodes), the available physical and virtual memory of the system is insufficient even for loading in the data associated with the network. So, the objective of the proposed approach is to reduce the size of problem.

This objective is met by partitioning the given network into sub networks of manageable size, and solving the network by solving the sub-pieces individually. Since the entire network is tightly connected, we cannot ignore the interaction between the various partitions. So, in order to account for the interactions between the partitions, while at the same time not enlarging the size of the problem at hand, we use models for the partitions that capture their behavior as observed at their interface nodes (also called ports). We refer to these models as macromodels. A macromodel is a multi-port linear circuit element that has the same linear relation between the voltages and currents through its ports as the partition itself. With macromodels for partitions available, the original (unpartitioned) network is efficiently solved after replacing the partitions by the respective macromodels, as the macromodels are much smaller in size than the partitions themselves.

This macromodeling approach provides a significant speedup as the creation of macromodels for the partitions can be performed in parallel.

2.2.1 HIERARCHICAL MODELING

The macromodel approach to power grid analysis is illustrated in figure 2.3. Let us consider a division of the entire power network into one global partition and k local partitions. A node in a local partition having links only to other nodes in the same partition is called an internal node, a node in the global partition is called a global node, and a node in a local partition that is connected to some node outside the local partition (i.e., in the global partition) is called a port. The global grid is then defined to include the set of nodes that lie in the global partition and the port nodes, while the grid in a local partition constitutes a local grid.

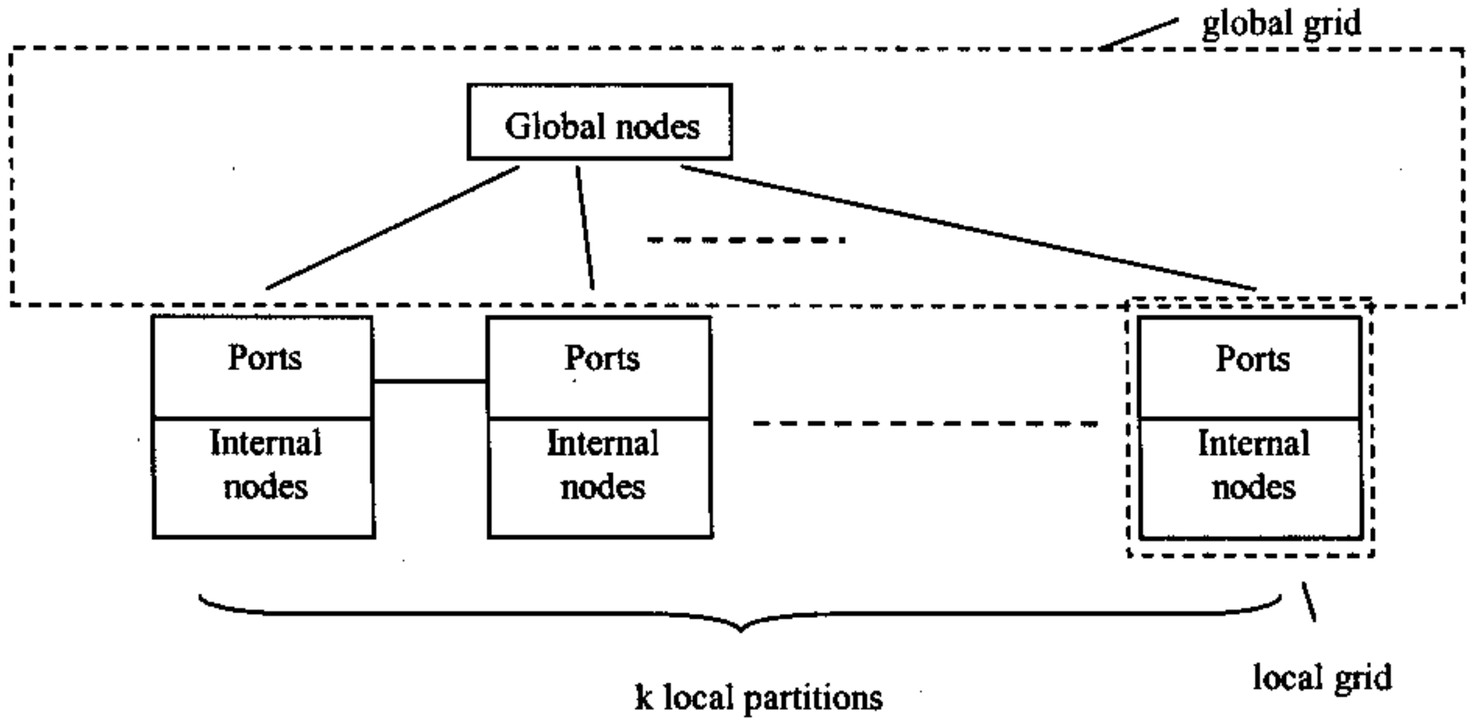


Figure 2.3: Hierarchical power network analysis

Now each of the k local grids can be modeled as a multi-port linear element with transfer characteristics given by

$$I = A.V+S, \text{ ------(3)}$$

where m is the number of ports in the local grid, A is the port admittance matrix, V is the vector of voltages at the ports, I is the current through the interface between the local and global grids, S is a vector of current sources connected between each port and reference node. We refer to set (A,S) as the macromodel of the respective local grid. The macromodel (A,S) in eqn (3) is obtained through a reduction procedure starting from the modified nodal equations of the local grid expressed in the form:

$$G.U = J, \text{ ---------(4)}$$

where n is number of nodes in the local grid, G is the coefficient matrix, U is the unknown vector of the nodes of the local grid, and J is vector of currents that flow out of each node in the local grid. For the port nodes, J would also include the currents through the interface between the local and global grids. The procedure of deriving the transfer characteristic in eqn (3) from the modified nodal equation (4) is referred as macromodeling, and described later.

Once the macromodels for all the local grids are generated, the entire network can be abstracted simply as the global grid, with the macromodel elements connected to it at the port nodes. This is achieved by combining the coefficient matrix and the RHS current vector of the global grid with the macromodels, (A,S); Equations (3) of

each local grid can be stamped into the modified nodal equations of the global grid. This will be a system of (n_0+m) linear equations, where n_0 is the number of global nodes and m is the number of ports in global grid.

From the above reduction scheme, the voltages and currents in the entire power grid can be solved in the following steps:

- Obtain global grid voltages by solving MNA eqns of global grid.
- For each partition, obtain I from eqn (3) using the port voltages.
- Solve eqn(4) for each partition using I on the right hand side, to obtain voltages at the internal nodes of partitions.

The flow of the macromodel approach is illustrated in following figure.

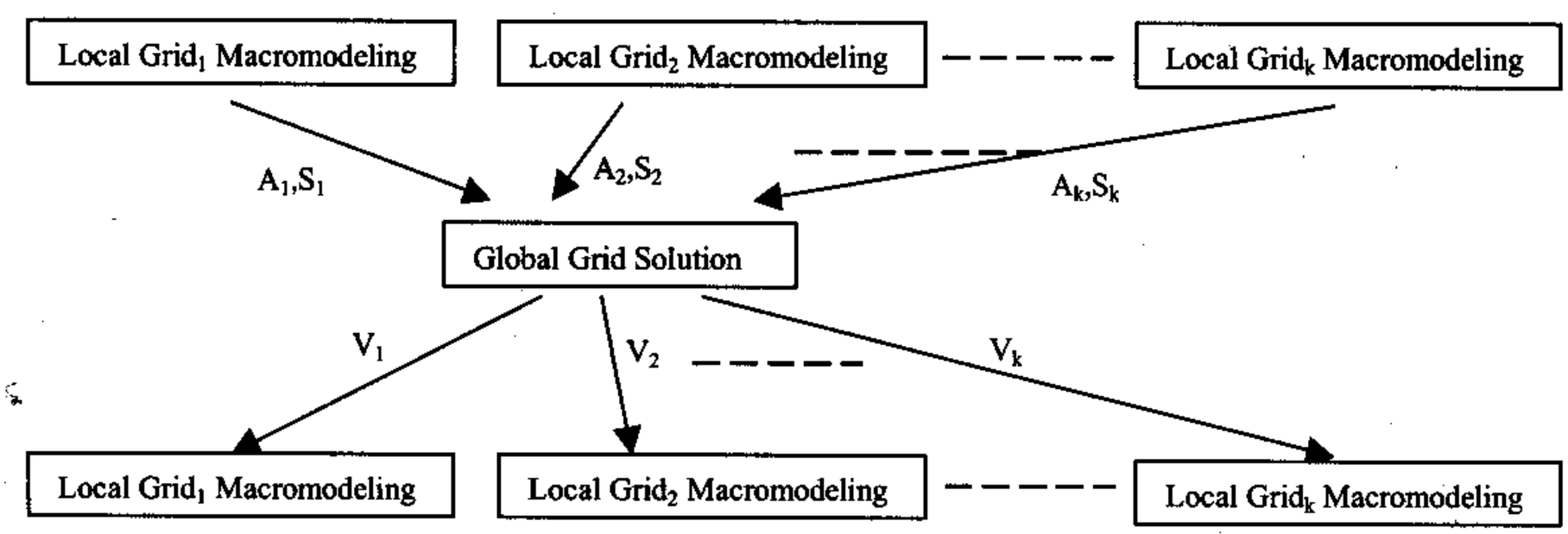


Figure 2.4: Flow of Macromodel

2.2.2 Macromodeling

Macromodeling is the procedure of deriving equation (3) from the modified nodal equations (4) of the partition. The modified nodal equations for a partition can be written as

$$\begin{bmatrix} G_{11} & G_{12} \\ G_{12}^T & G_{22} \end{bmatrix} \begin{bmatrix} U_1 \\ V \end{bmatrix} = \begin{bmatrix} J_1 \\ J_2+I \end{bmatrix} \qquad \text{------- (5)}$$

where

- U_1 and V are vectors of voltages at the internal nodes and ports respectively
- J_1 and J_2 are vectors of current sources connected at the internal nodes and ports respectively.

- I is the vector of currents through the interface.
- G_{12} is the admittance of links between the internal nodes and the ports
- G_{11} is the admittance matrix of internal nodes
- G_{22} is the admittance matrix of ports.

From (5), we may rewrite the first set of equations as

$$U_1 = (G_{11})^{-1}(J_1 - G_{12}V) \quad \text{---------- (6)}$$

Substituting this value of U_1 into the second equation of (5), we get

$$I = (G_{22} - G_{12}^{T}G_{11}^{-1}G_{12})V + (G_{12}^{T}G_{11}^{-1}J_1 - J_2) \quad \text{------- (7)}$$

Here $(G_{22} - G_{12}^{T}G_{11}^{-1}G_{12})$ is the port admittance matrix in eqn (3) and $(G_{12}^{T}G_{11}^{-1}J_1 - J_2)$ is the constant vector S in eqn (3).

2.3 Closed Form Analysis of Circuit

Once we formulated the circuit equations using MNA, we can find the nodal voltages and currents in the circuit at different timestamps by iteratively solving them. Usually, in most of the cases, we want the nodal voltages and currents after particular number of time steps say 'n'. For this, we need to iteratively solve the MNA equations n times. But, by the closed form analysis of circuit, we can overcome this problem by directly finding the nodal voltages and currents in the circuit after n time steps. In the following, we present a method to do closed form analysis with some assumptions.

Using MNA, we get the circuit equations in the form of AX = B where X is the unknown vector of nodal voltages and currents. Once the matrix A is formed, it is constant through out the iterations. Matrix A depends on the given circuit and the time step. But vector B depends on the present time point voltage source and current source values and previous time point unknown vector. This can be inferred from the formation of MNA equations.

In our power grid simulation, we use the DC voltage source and DC or AC current sources. Let us assume that we have all our current sources in our model as DC. Then, at any time step, vector B depends on the previous time step unknown vector and have some constant added to it that depends on the voltage and current sources.

Initially, we obtained matrix A and vector B_1 by using MNA method. On solving $AX_1 = B_1$, we got X_1. At next time step, matrix A will not be changed but rhs i.e., vector B will be changed. For next time step, rhs can be written as

$$B_2 = K + V*X_1 \quad \text{----------} \quad (8)$$

where K is vector which depends on voltage and current sources and V is matrix which depends on capacitors and inductors. It is advantageous to consider the contributions of each circuit element to the V matrix and K vector separately. The "element rubber stamps" given in Table II corresponding to the circuit of a "general node" shown in Fig 2.1 summarize the contributions for each type of element, where BR refers to the additional branch relation. The matrix V and vector K can be generated by using the element stamp table in a straightforward manner. For a given circuit, the matrix V dimension is same as that of the matrix A and vector K dimension is same as that of vector B formed by MNA. The construction of matrix V and vector K can be done along with that of matrix A formed by MNA. For next time step, the unknown vector can be written as

$$X_2 = A^{-1}(K+V*X_1) \quad \text{----} \quad (9)$$

Stamp table for Construction of matrix V:

		Current not output			Branch Current output			
C		Vi	V3		Vi	V3	IC
	i	c/h	-c/h	i			
	3	-c/h	c/h	3			
				BR	c/h	-c/h	
L				Vi	V5	IL
		L		i			
				5			
				BR			-L/h

Stamp table for Construction of vector K:

E		E		K	
			i		
			4		
			BR	E	
J		K		K	
	i	-J	i		
	2	J	2		
			BR	J	

Table II

Following the same lines as above, we can write n^{th} time step rhs as

$$B_n = K + V*X_{(n-1)} \quad \text{---- (10)}$$

and unknown vectors at successive time steps as

$$X_3 = A^{-1}(K+V*X_2)$$

$$= A^{-1}K + A^{-1}V\,A^{-1}K + A^{-1}V\,A^{-1}VX_1 \text{ (from eqn 9)}$$

$$X_3 = (I+ A^{-1}V)\,A^{-1}K + (A^{-1}V)^2X_1 \quad \text{---- (11)}$$

where I is the identity matrix of dimension same to that of matrix A.

$$X_4 = A^{-1}(K+V*X_3) \quad \text{---- (12)}$$

From eqns (11) and (12), we can write

$$X_4 = (I + A^{-1}V + (A^{-1}V)^2)\,A^{-1}K + (A^{-1}V)^3X_1 \text{ ---- (13)}$$

From eqn (13), we can write $X_5 = (I+ A^{-1}V + (A^{-1}V)^2 + (A^{-1}V)^3)\,A^{-1}K + (A^{-1}V)^4X_1$

On generalizing the above equations, we can write unknown vector at n^{th} time step as

$$X_n = (I+ A^{-1}V + (A^{-1}V)^2 + \text{---} + (A^{-1}V)^{(n-2)})\,A^{-1}K + (A^{-1}V)^{(n-1)}X_1$$

$$X_n = ((A^{-1}V)^{(n-1)} - I)\,(A^{-1}V - I)^{-1}\,A^{-1}K + (A^{-1}V)^{(n-1)}X_1 \text{ ---- (14)}$$

Using above result, we can find n^{th} time step unknown vector without finding intermediate time point unknown vectors.

For the circuit in figure 2.2, the following matrix V is constructed from stamp table II.

	1	2	3	4	5	6	7	IL1	IL2	IE	IJ
1	0	0	0	0	0	0	0	0	0	0	0
2	0	C1/h	-C1/h	0	0	0	0	0	0	0	0
3	0	-C1/h	C1/h	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	C2/h	-C2/h	0	0	0	0	0
6	0	0	0	0	-C2/h	C2/h	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0
IL1	0	0	0	0	0	0	0	-L1/h	0	0	0
IL2	0	0	0	0	0	0	0	0	-L2/h	0	0
IE	0	0	0	0	0	0	0	0	0	0	0
IJ	0	0	0	0	0	0	0	0	0	0	0

Similarly, the vector K is constructed from the stamp table II for the circuit in figure 2.2.

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ IL1 \\ IL2 \\ IE \\ IJ \end{array} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ E \\ I \end{pmatrix}$$

Using the above matrix V and vector K, we can construct the next time step RHS matrix using the previous unknown vector, as given in equation (10) and the n^{th} step solution using the equation (14).

3. Simulation and Results

Modern integrated circuit network includes millions of nodes and tens of millions of elements. For effectively analyzing such a power grid, the simulation technique used should be efficient. The above-mentioned techniques are efficient in their own way and had been implemented.

The equivalent electric circuit of power grid considered for simulation is as follows:

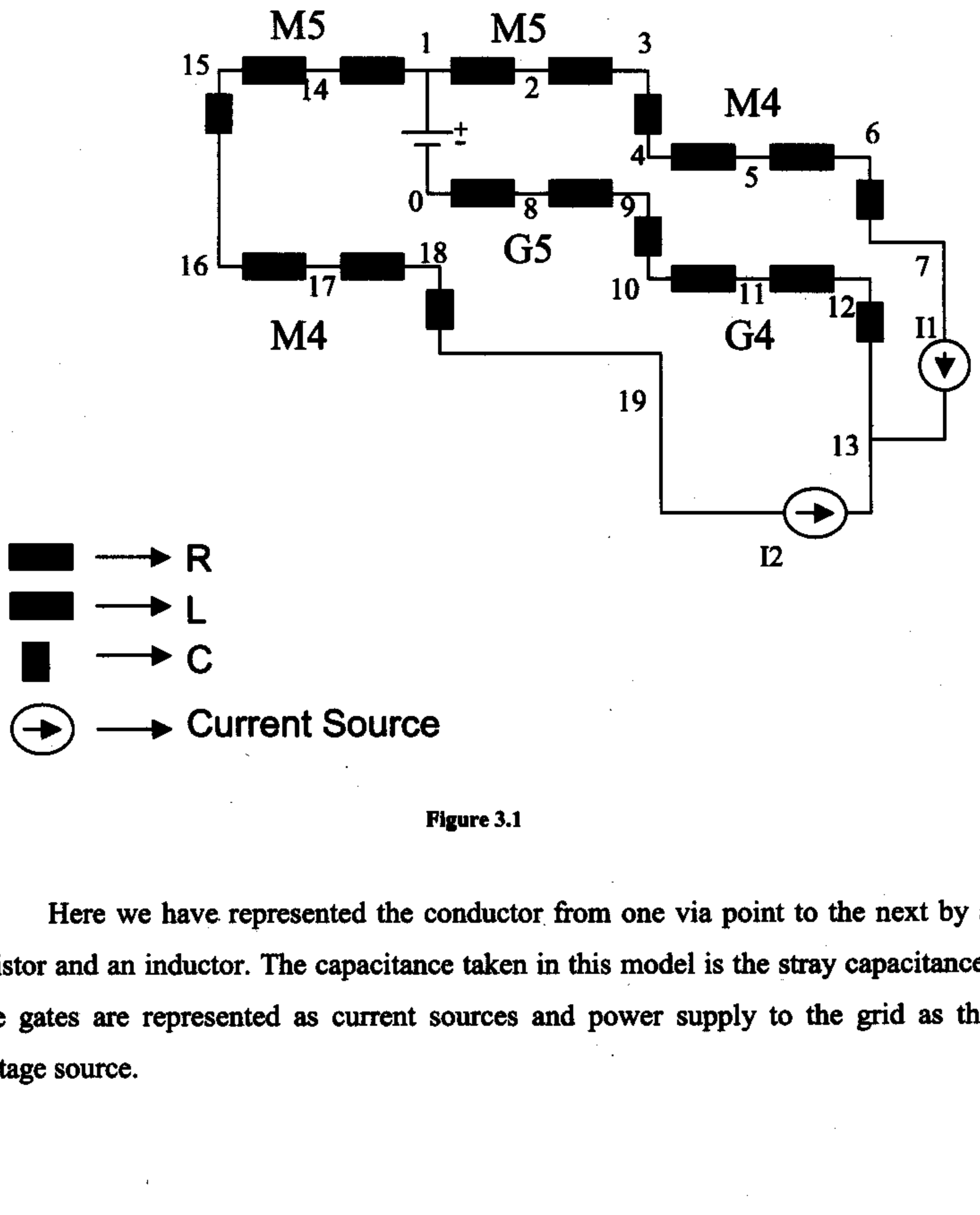


Figure 3.1

Here we have represented the conductor from one via point to the next by a resistor and an inductor. The capacitance taken in this model is the stray capacitance. The gates are represented as current sources and power supply to the grid as the voltage source.

3.1 Modified Nodal Analysis:

We had simulated Modified Nodal Analysis technique in C, Mathematica. The input structure to the simulation is very much nearer to the PSPICE input format of the circuit and we term it as MNA format. The part of input format of the considered circuit is as follows:

r_1	2	1	52.17e-3									
l_1	3	2	0.354e-12									
c_1	4	3	0.169e-15									
v_1	1	0	1.2									
i_1	8	7	SIN	2	1e-9	6.28						
i_2	19	8	PER	8	0	0	1e-9	1.1e-9	2e-9	1.1e-9	3e-9	0

General format of a line in input file to this simulation is as follows:

<character><num> <node 1> <node2> <(String, set of values) Or value>

In above, "character" can be 'r' to represent resistor, or 'c' to represent capacitor, or 'l' to represent inductor, or 'v' to represent voltage source, or 'i' to represent current source.

"num" can be any positive number to differentiate that circuit element from other similar circuit elements. Thus character along with num corresponds to some circuit element in network.

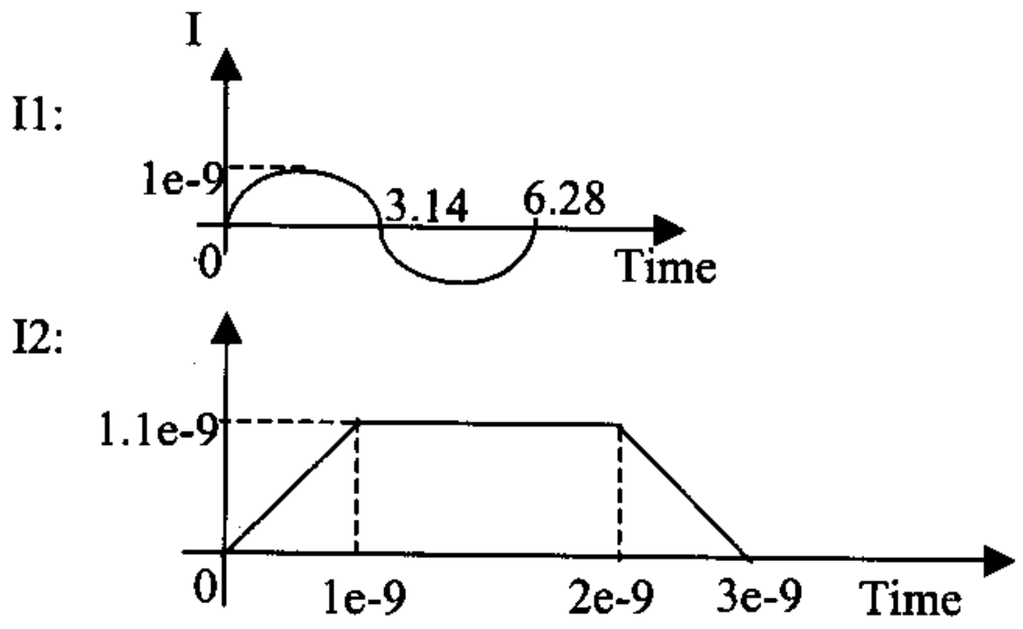
"node1" and "node2" are two nodes in the circuit, where that corresponding circuit element is present in the network. Also, current is assumed to flow from node1 to node2 in the circuit through that element.

If character is 'r' or 'c' or 'l' or 'v', the last parameter will be a single value, and it gives the conductance or capacitance or inductance or potential difference of that circuit element respectively depending on its character.

In above input, first line says that r1 is the conductor between the nodes 1 and 2, the value of conductance is 52.17e-3 and the direction of current is assumed to be from node 2 to node 1. Similarly, second line says that l1 is the inductor between the nodes 2 and 3, its inductance value is 0.354e-12 and the direction of current is assumed to be from node 3 to node 2.

If character is 'i', then string gives the type of waveform of that current element and the set of values gives the parameters of the waveform.

In above circuit, the current source i_1 is modeled as the SIN waveform and current source i_2 is modeled as the PERIODIC waveform. Any other type of current sources can be added to simulation. Current waveforms for above statements in input file are plotted in below diagram:



ALGORITHM

Input: Txt file in which circuit information is given as specified above and initial point voltages and currents.

Output: Nodal voltages and branch currents at each time step.

Step 1: Read the input file and group the information about resistors, conductors, inductors, voltage sources and current sources.

Step 2: Construct the matrix A & vector B using MODIFIED NODAL ANALYSIS method.

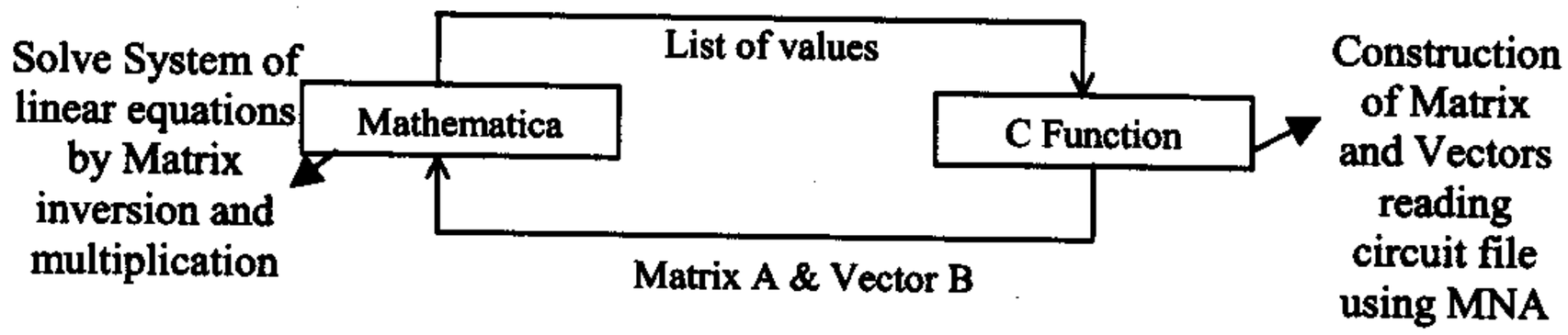
Step 3: X can be calculated by $A^{-1}B$.

[X is present time point nodal voltages and currents in circuit]

Step 4: Exit if required time point unknown vector is found else Go to Step 2 to find voltages for next time point and use X, while constructing matrix B in step 2.

First 2 steps of algorithm are implemented in C and third step is done in mathematica.

The **flow** in the MNA approach is illustrated in following figure:



Results:

In order to crosscheck our results, we considered a circuit from SPICE text book [6] and simulated it with our programs. The circuit which we considered from SPICE is as follows:

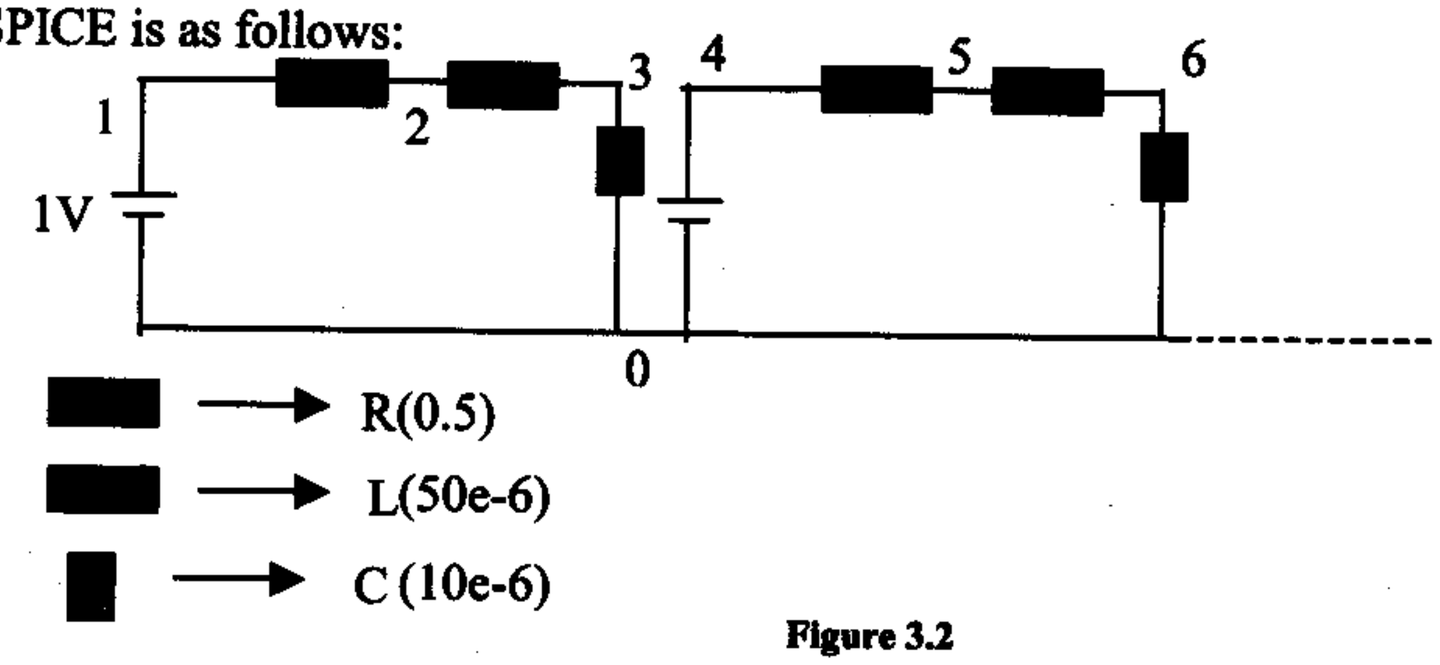


Figure 3.2

Nodal voltages and currents of our simulation and PSPICE are well matched and the below table gives the real times used by C & Mathematica program for different sizes of network of above figure and for different number of iterations:

# of elements	Res. Matrix size	Time (sec)	Iterations
1000R, 1000C, 1000L, 1000V	5000	90	50
1000R, 1000C, 1000L, 1000V	5000	60	2
200R, 200C, 200L, 200V	1000	180	5000
1300R, 1300L, 1300C, 1300V	6500	100	1
1350R, 1350L, 1350C,1350V	6750	140	50
1400R, 1400L, 1400C, 1400V	7000	Not handled	

Table 3.1: Real Times used by C and Mathematica

In above table, # of elements gives the count of each type of circuit element in circuit. Res. Matrix size is the total number of MNA equations formed, i.e., the size of matrix to be handled. Time is the time taken to solve the matrix, iterations is the number of times the MNA equations are iteratively solved.

If we use C for solving system of linear equations, then we can handle more matrix sizes than that handled by C and Mathematica. C can handle matrices of size upto 15000. Most of the time specified above is used by Mathematica rather than by C.

Following table gives the real times used by C simulation for different sizes of above circuit:

# of elements	Res. Matrix size	Time (min)	Iterations
1000R, 1000C, 1000L, 1000V	5000	47	4
1400R, 1400L, 1400C, 1400V	7000	127	4
2000R, 2000L, 2000C, 2000V	10000	660	4
3000R, 3000L, 3000C, 3000V	15000	Not handled	

Table 3.2: Real times used by C

The huge difference in real times of above 2 simulations is because Mathematica is using parallel computing and there are 4 processors in the system in which execution is done. These results show the advantages of cluster environment computing. Number of iterations doesn't come as parameter until it is comparable to matrix size.

So, if the size of matrix can be handled by Mathematica, then we can use the simulation done by using C and Mathematica. Otherwise, we can use the simulation done by using C. In our simulation with C, we used LU Decomposition with partial pivoting to solve system of linear equations. Efficient methodology of solving linear equations either serially or parallel, keeping accuracy, can decrease the time spent by C simulation.

As our purpose is power grid simulation, we simulated the power grid shown in fig 3.1 with 3 metal layers for different sizes. Following table gives the real times used by C simulation for different sizes of our concerned power grid circuit:

# of elements	Res. Matrix size	Time (min)	Iterations
1200R, 1200C, 1200L, 200I	5000	86	4
1680R, 1680L, 1680C, 280I	7000	233	4
2400R, 2400L, 2400C, 400I	10000	680	4
3600R, 3600L, 3600C, 600I	15000	Not handled	

Table 3.3: Real times used by C

3.2 Hierarchical Analysis:

This technique had been implemented in C++. The input to this simulation is MNA format of each partition of the electrical circuit added with some labels. The equivalent circuit of power grid considered for simulating is partitioned as below:

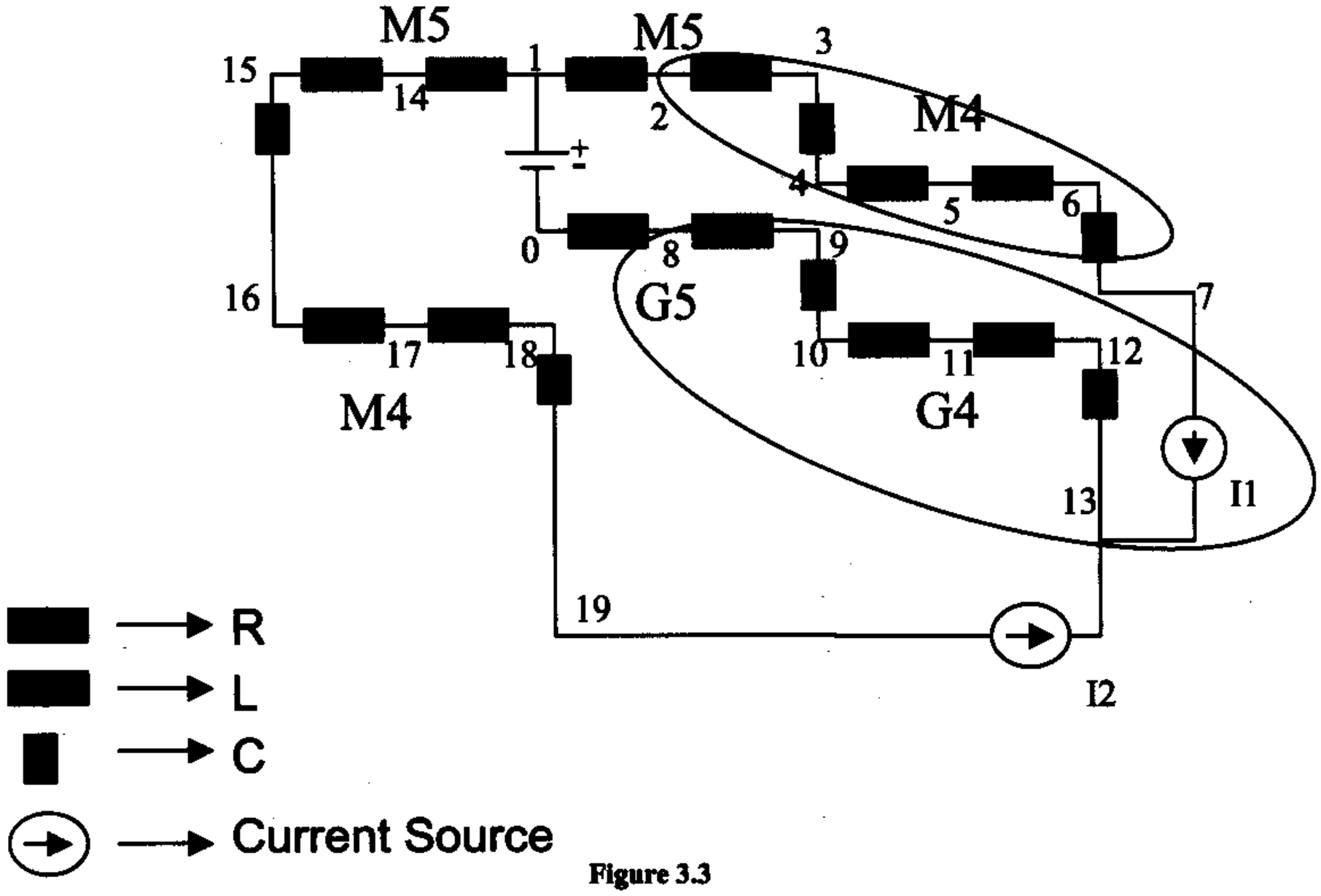


Figure 3.3

The part of input format of the above circuit is as follows:

```
2
2       7
l1      2       3       0.354e-12
c1      3       4       0.169e-15
r2      4       5       52.17e-3
l2      5       6       0.354e-12
c2      6       7       0.169e-15
END_LOCAL
```

The input file of this simulation consists of information about each local grid separately. General format of a local grid is as follows:

<Number of ports>

<Port numbers>

<MNA format of the local grid>

END_LOCAL

First line gives us the number of ports in the local grid and second line gives us the port numbers. In above case, there are 2 ports and they are 2 and 7. After listing the circuit elements in the local grid, it should be followed by END_LOCAL label indicating the end of local grid.

Once the information about local grid is completed, it is followed by information about another local grid in the same format. At the last, the global grid information will be augmented. Information of global grid also needs to have number of ports, port numbers and MNA format of global grid, but it does not contain the label "END_LOCAL". Thus, completes the input file format.

ALGORITHM:

Input: Txt file in which circuit information is given as specified above and initial time point voltages and currents.

Output: Nodal voltages and branch currents at each time step.

Step 1: Read the input file and group information about resistors, capacitances, inductors, current sources and voltage sources of a partition.

Step 2: Construct matrices A and B using MNA method.

Step 3: Obtain (A_i,S_i) macromodel of partition from A and B matrices constructed above using Macromodeling approach and write them into temporary file say models.txt. Also write partially solved equations into another temporary file say partialsoln.txt.

Step 4: Repeat the above 3 steps for all partitions until global partition is reached in input file.

Step 5: Read the global partition information from input file and macromodels from temporary file temp and construct A & B matrices as specified in Hierarchical Modeling.

Step 6: Obtain nodal voltages and branch currents in global grid and ports by solving system of linear equations formed in step 5.

Step 7: Backtrack the partially solved equations of step 3 using the solution obtained in step 6 to obtain the overall power grid solution.

Step 8: Write the global grid and local grids solutions into another temporary file say output.txt for the next time point usage.

Step 9: Repeat above 8 steps until required time point has been reached.

Results:

To analyze the circuit in fig 3.2 using hierarchical method, it has been partitioned as follows:

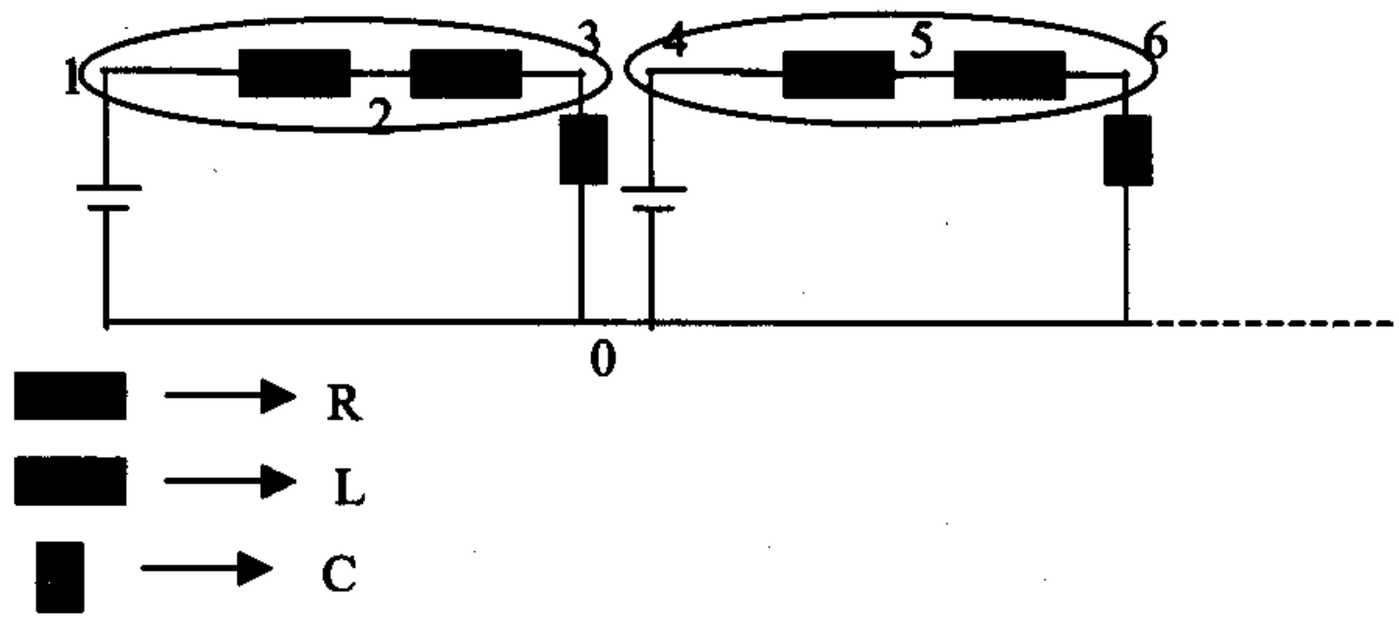


Figure 3.4

The following table gives the real time used by the simulation done using hierarchical technique on the circuit shown in fig 3.4:

# of elements	Before Reduction, Matrix Size	Max Res. Matrix size	Time(Before(min), After(min))
1000R, 1000C, 1000L, 1000V	5000	3000	(47,10)
1400R, 1400L, 1400C, 1400V	7000	4200	(127,27.2)
3000R, 3000L, 3000C, 3000V	15000	9000	(Not handle,600)

Table 3.4: Real times used by Hierarchical simulation

In above table, "Before reduction, matrix size" is the size of matrix formed in case of traditional MNA. "Max Res. Matrix size" is the maximum of the partition sizes in hierarchical analysis on which execution time or storage requirement depends. Time gives the execution time taken by traditional MNA and Hierarchical analysis.

Traditional simulation of MNA results shows that it cannot handle circuits that results matrices of sizes greater than 15000. But hierarchical simulation results shows that it can be handled with this technique.

The real time used by hierarchical simulation mainly depends on the maximum size of all the grids. Proper partitioning of circuit will improve the execution time.

Simulation results of the circuit shown in fig 3.3 are given in the following table and the results shows the effect of partitioning:

# of elements	Before Reduction, Matrix Size	Max Res. Matrix size	Time(Before(min), After(sec))
1200R, 1200C, 1200L, 200I	5000	600	(86,8)
2400R, 2400L, 2400C, 400I	10000	1200	(680,64)
3600R, 3600L, 3600C, 600I	15000	1800	(Not handle,220)

Table 3.5: Real times used by Hierarchical simulation

Hierarchical simulation can handle any # of elements but the maximum size of the matrices resulted should not cross 15000. As specified in the algorithm, in hierarchical analysis, we use the files and so limit of file size also should be considered. The results shown above are for one iteration. Unlike to traditional MNA, hierarchical takes same time for any successive iteration. So, time required increases linearly with the number of iterations unlike to traditional MNA.

3.3 Closed Form analysis:

This method had been implemented in C. The implementation is similar to that of MNA technique. Here the construction of matrices V and K play important role.

ALGORITHM

Input: Txt file in which circuit information is given in MNA format, 'n' time step for which unknown vector to be found and initial time point voltages and currents.

Output: Nodal voltages and branch currents at n^{th} time step.

Step 1: Read the input file and group the information about resistors, conductors, inductors, voltage sources and current sources.

Step 2: Construct the matrices A, V, K & B using closed formed analysis method.

Step 3: X_1 can be calculated by $A^{-1}B$.

[X_1 is first time point nodal voltages and currents in circuit]

Step 4: Find $(A^{-1}V)^{(n-1)}$ using repeated square method.

Step 5: Using closed form analysis result, n^{th} time step unknown vector can be found.

Results:

Time complexity of the above algorithm is $O(\max((size)^3,(size)^{2.7}\log(n)))$ where size is the size of the matrix A and n is the number of iterations and that of traditional MNA is $O(\max((size)^3,(size)^2 n))$. So depending on the problem, we can select one of these methods. If we need to do more number of iterations for smaller matrix sizes, better if we use this closed form analysis. So, if matrix size is comparably larger to that of number of iterations, then closed form analysis takes less time compared to that of traditional MNA.

The same can be shown by the following table. Execution time of simulation of closed form technique against C, Mathematica simulation on different sizes of network of fig 3.2 and for different iterations is given in the following table:

# of elements	Res. Matrix size	Time (sec) (MNA, Closed)	Iterations
100R, 100C, 100L, 100V	500	(3, 70)	1024
50R, 50L, 50C, 50V	250	(2,8)	1024
50R, 50L, 50C, 50V	250	(22, 12)	32768
50R, 50L, 50C, 50V	250	(781,16)	1000000

Table 3.6: Real times used by closed form analysis

Execution time of simulation of closed form technique on fig 3.1 for different iterations is given in following table:

# of elements	Res. Matrix size	Time (sec) (MNA, Closed)	Iterations
120R, 120C, 120L, 20I	500	(9, 116)	1024
60R, 60L, 60C, 20I	250	(2,14)	1024
60R, 60L, 60C, 20I	250	(33, 20)	32768
60R, 60L, 60C, 20I	250	(1025,25)	1000000

Table 3.7: Real times used by closed form analysis

The disadvantage of hierarchical method over traditional MNA, i.e., time increases linearly with number of iterations, can be overcome if we combine hierarchical method with closed form analysis.

An issue in the closed form analysis is the accuracy. As the number of iterations increases, the accuracy of the result gets worsened.

3.4 Comparison of the above three approaches:

We have already mentioned that, the method by which circuit equations are formulated affects the storage requirements, execution time of the program and the accuracy of the result.

In this sub-section, we compare the above three approaches with these three parameters.

In terms of storage requirement, C simulation of traditional MNA takes double the storage needed than that by C and Mathematica simulation, since, in C simulation, in order to find matrix inverse, we need extra memory. However, C and Mathematica cannot handle large circuits.

Hierarchical analysis takes much less space compared to any other simulation, as the storage needs depends on the maximum of the partition sizes.

Closed form analysis takes double the storage that needed by the C simulation of traditional MNA, as we need to find matrices V and $(A^{-1}V - I)^{-1}$, which requires more memory.

C & Mathematica simulation took very less execution time compared to other simulations, because of parallel computation by Mathematica. But as the number of iterations increases, closed form analysis will perform faster.

However, hierarchical analysis performs better than C simulation of traditional MNA, given that partitioning is properly done.

C and mathematica simulation will give better accuracy than any other simulation as mathematica can handle even smaller values than that handled by C or C++. C or C++ can give accuracy up to 10^{-9}.

Due to repetitive multiplication of matrices in closed form analysis, it gives accuracy only upto 10^{-2} for 32 iterations.

The following are the experimental results:

Execution time and storage requirement comparison between traditional MNA and hierarchical analysis is as follows:

Circuit Elements	Traditional MNA (size,time,mem)		Hierarchical Analysis (C++) (size,time,mem)
	C, Mathematica	C	
1000R, 1000L, 1000C, 1000V	5000, 90 sec, 25MB	5000, 47 min, 50 MB	3000,10 min, 18MB
1400R, 1400L, 1400C, 1400V	Not Handled	7000,127 min, 100MB	4200, 27.2 min, 32MB
3000R, 3000L, 3000C, 3000V	NH	NH	9000,600 min, 160MB

Table 3.8: Real times and storage area used by traditional MNA and hierarchical analysis

Each entry in above table gives the max size of matrix to be handled, time and storage area required to execute networks of those number of elements. Size with the hierarchical analysis depends on the partitioning style of that circuit.

Traditional MNA by C simulation and closed form analysis comparison is as follows:

Size	Closed Form Analysis (time, mem)	MNA (time,mem)	Iterations
500	(116,500KB)	(9,250KB)	1024
250	(14,250KB)	(2,125KB)	1024
250	(20,250KB)	(33,125KB)	32768
250	(25,250KB)	(1025,125KB)	1000000

Table 3.9: Real times and storage area used by traditional MNA and closed form analysis

But, in closed form analysis, as iterations increases, the accuracy of the values get worsened. As already specified, for 32 iterations, it gives accuracy only up to 10^{-2}.

4. CONCLUSION

The problem of power dissipation in the power delivery network is one of the challenges in nanometer technology. The considered equivalent RLC model of power grid had been simulated by the various techniques discussed above.

MNA approach is the basic method for formulating circuit equations. The hierarchical modeling approach is alternative to traditional nonhierarchical analysis method, capable of handling the increasing size of power grids. Closed form analysis is totally new of its kind, used when the intermediate time points nodal voltages and branch currents are not required.

All three had been simulated and results are presented as above.

FUTURE SCOPE

Most of the time in power grid analysis, there is no need to find all nodal voltages and branch currents. For this, grid reduction technique can be added into simulation.

Using sparse techniques, we can improve run time of the program. Also, closed form analysis can be applied to hierarchical modeling whenever intermediate time points nodal voltages and branch currents are not required.

Accuracy of the calculations can be improved.

In Hierarchical modeling, we can explore optimal partitioning techniques that can be applied with minimum user-intervention.

Closed form analysis can be extended to AC waveforms, as well as to DC, keeping error within bounds and increasing the accuracy. Also, closed form analysis can be combined with hierarchical analysis to overcome the disadvantage of hierarchical analysis over traditional MNA.

As there is lot of scope for parallel computation in all the above analyses, implementing it will improve the run time of the program.

5. References:

[1] Susmita Sur Kolay, Chandra Tirumurti, Sandip Kundu, Yi-Shing Chang, and Sujit Zachariah. "Analysis and Modeling of Power Supply Related Failures"

[2] C. Ho, A. Ruehli, and P.Brennan. "The modified nodal approach to network analysis," *IEEE Trans. Circuits and System*, vol. CAS-22, no6, pp. 504-509, 1975.

[3] Min Zhao, Rajendran V. Panda, Sachin S. Sapatnekar, Tim Edwards, Rajat Chaudhry, and David Blaauw. "Hierarchical Analysis of Power Distribution Networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21,no.2, pp.159-168, Feb. 2002.

[4] Haifeng Qian, Sani R. Nassif, and Sachin S. Sapatnekar. "Random walks in a Supply Network"

[5] Sani R. Nassif and Joseph N. Kozhaya. "Fast Power Grid Simulation"

[6] SPICE for Circuits and Electronics using PSPICE by Muhammad H. Rashid.

[7] Network Analysis by M.E. Van Valkenburg