2004

Estimation of switching activity in sequential circuits using dynamic Bayesian Networks

Karthikeyan Lingasubramanian
University of South Florida

Estimation of Switching Activity in Sequential Circuits using Dynamic Bayesian Networks

by

Karthikeyan Lingasubramanian

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Electrical Engineering
Department of Electrical Engineering
College of Engineering
University of South Florida

Major Professor: Sanjukta Bhanja, Ph.D.
Nagarajan Ranganathan, Ph.D.
Wilfrido A. Moreno, Ph.D.

Date of Approval:
June 2, 2004

Keywords: Probabilistic model, Simulation, Clique, Inference, Sampling

DEDICATION

To my loving Father and Mother

ACKNOWLEDGEMENTS

I would like to take this golden opportunity to thank my major professor Dr. Sanjukta Bhanja. Without her this work wouldnt have been possible. She supported me and helped me to the core. She gave me complete freedom in research prospective. She has helped me a lot to mould myself as a researcher. She has trained me in every aspect of research like reading, writing etc. Moreover she has also been a good friend to me.

My sincere thanks to Dr. Nagarajan Ranganathan and Dr. Wilfredo A. Moreno for serving in my committee.

I am really very grateful for the invaluable support and motivation that I recieved from my family.

I would also like to thank all my friends for their able support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ESTIMATION OF SWITCHING ACTIVITY IN SEQUENTIAL CIRCUITS USING DYNAMIC BAYESIAN NETWORKS

Karthikeyan Lingasubramanian

ABSTRACT

This thesis presents a novel, non-simulative, probabilistic model for switching activity in sequential circuits, capturing both spatio-temporal correlations at internal nodes and higher order temporal correlations due to feedback. Switching activity, one of the key component in dynamic power dissipation, is dependent on input streams and exhibits spatio-temporal correlation amongst the signals. One can handle dependency modeling of switching activity in a combinational circuit by Bayesian Networks [2] that encapsulates the underlying joint probability distribution function exactly.

We present the underlying switching model of a sequential circuit as the time coupled logic induced directed acyclic graph (TC-LiDAG), that can be constructed from the logic structure and prove it to be a dynamic Bayesian Network. Dynamic Bayesian Networks over *n* time slices are also minimal representation of the dependency model where nodes denote the random variable and edges either denote direct dependency between variables at one time instant or denote dependencies between the random variables at different time instants. Dynamic Bayesian Networks are extremely powerful in modeling higher order temporal as well as spatial correlations; it is an exact model for the underlying conditional independencies. The attractive feature of this graphical representation of the joint probability function is that not only does it make the dependency relationships amongst the nodes explicit but it also serves as a computational mechanism for probabilistic inference.

We use stochastic inference engines for dynamic Bayesian Networks which provides any-time estimates and scales well with respect to size We observe that less than a thousand samples usually converge to the correct estimates and that three time slices are sufficient for the ISCAS benchmark circuits. The average errors in switching probability of 0.006, with errors tightly distributed around the mean error values, on ISCAS'89 benchmark circuits involving up to 10000 signals are reported.

CHAPTER 1

INTRODUCTION

The ability to form accurate estimates of power usage, both dynamic and static, of VLSI circuits is an important issue for rapid design-space exploration. Switching activity is one important component in dynamic power dissipation that is independent of the technology of the implementation of the VLSI circuit. Contribution to total power due to switching is dependent on the logic of the circuit and the inputs and will be present even if sizes of circuits reduce to nano domain. Apart from contributing to power, switching in circuits is also important from reliability point of view and hence can be considered to be fundamental in capturing the dynamic aspects of VLSI circuits.

Among different types of VLSI circuits, switching in sequential circuits, which also happens to be the most common type of logic, are the hardest to estimate. This is particularly due to the complex higher order dependencies in the switching profile, induced by the spatio-temporal components of the main circuit but mainly caused by the state feedbacks that are present. These state feedbacks are not present in pure combinational circuits. One important aspect of switching dependencies in sequential circuits that one can exploit is the first order Markov property, i.e. the system state is independent of all past states given just the previous state. This is true because the dependencies are ultimately created due to logic and re-convergence, using just the current and last values.

The complexity of switching in sequential circuits arise due to the presence of feedback in basic components such as flip-flops and latches. The inputs to a sequential circuit are not only the primary inputs but also these feedback signals. The feedback lines can be looked upon as determining the state of the circuits at each time instant. The state probabilities affect the state feedback line probabilities that, in turn, affect the switching probabilities in the entire circuit. Thus, formally, given a set of inputs i_t at a clock pulse and present states s_t, the next state signal s_{t+1} is uniquely determined as a function of i_t and s_t. At the next clock pulse, we have a new set of inputs i_{t+1} along with state s_{t+1} as an input to the circuit to obtain the next state signal s_{t+2}, and so on. Hence, the statistics of *both* spatial and

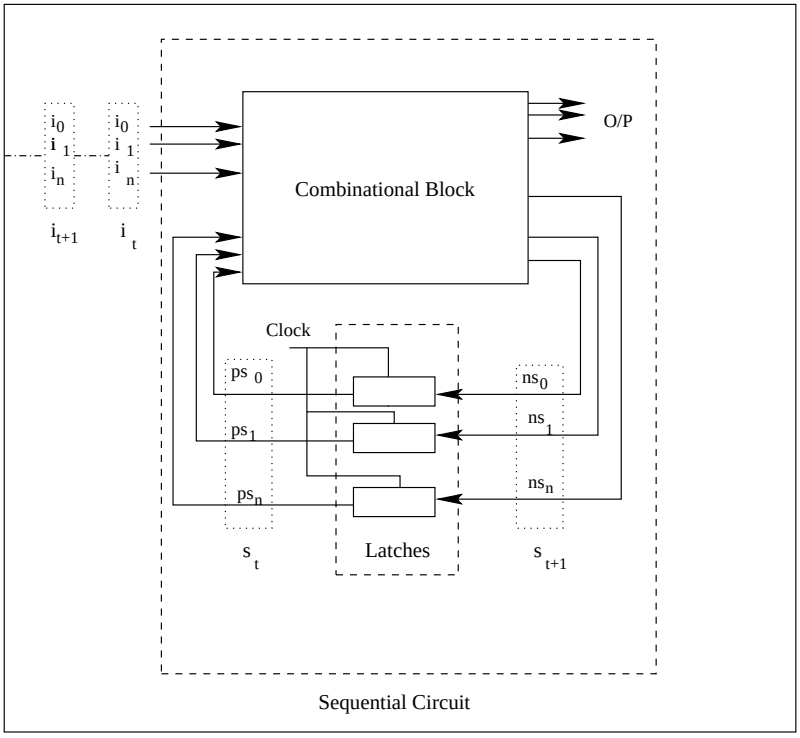


Figure 1.1. A Model for Sequential Circuit.

temporal correlations at the state lines are of great interest. It is important to be able to model both kinds of dependencies in these lines.

Previous works [1, 2, 3] have shown that switching activity in a combinational circuit can be *exactly* expressed in a probabilistic Bayesian Network, whose structure is a logic induced directed acyclic graph (LiDAG). Such models capture both the temporal and spatial dependencies in a compact manner using conditional probability specifications. For combinational circuits, first order temporal models are sufficient to completely capture dependencies under zero-delay. The attractive feature of the graphical representation of the joint probability distribution is that not only does it make the conditional dependencies among the nodes explicit, but it also serves as a computational mechanism for efficient probabilistic updating. The LiDAG structure, however, cannot model cyclical logical structure, like those induced by the feedback lines. This cyclic dependence effects the state line probabilities that, in turn, effect the switching probabilities in the entire circuit.

In this work, we propose a probabilistic, non-simulative, predictive model of the switching in sequential circuits using time coupled logic induced DAG (TC-LiDAG) structure that explicitly models

the higher order temporal and spatial dependencies among the feedback lines. This is achieved by coupling LiDAG representations of the combinational circuit from multiple time slices. The nodes in TC-LiDAG represent switching random variable at the primary input, state feedback, and internal lines. These random variables defined over four states, representing four possible signal transitions at each line which are $(0 \rightarrow 0, 0 \rightarrow 1, 1 \rightarrow 0, 1 \rightarrow 1)$. Edges of TC-LiDAG denote direct dependency. Some of them are dependencies within one time slice and the conditional probability specification for these are the same as in LiDAG, i.e. they capture the conditional probability of switching at an output line of a gate given the switching at the input lines of that gate. Rest of the edges are temporal, i.e. the edges are between nodes from different time slices, capturing the state dependencies between two consecutive time slices. We add another set of temporal edges between the same input line at two consecutive slices, capturing the implicit spatio-temporal dependencies in the input switchings. Temporal edges between just consecutive slices are sufficient because of the first order Markov property of the underlying logic.

We prove that the TC-LiDAG structure is a *Dynamic Bayesian Networks* (DBN) capturing all spatial and higher order temporal dependencies among the switchings in a sequential circuit. It is a

minimal representation, exploiting all the independencies. The model, in essence, builds a factored representation of the joint probability distribution of the switchings at all the lines in the circuit. Dynamic Bayesian Networks are extremely powerful graphical probabilistic models that encapsulates Hidden Markov Model (HMM) and Linear Dynamic System (LDS) and have been employed for gene matching [25], speech processing [24] and object tracking but their use in modeling VLSI events is new. Dynamic Bayesian Networks are basically Bayesian Networks defined over multiple time slices. The salient features of using a Dynamic Bayesian Networks are listed below:

1. DBN exploits conditional independence amongst the random variables of interest for factorization and reduction of representation of the underlying joint probability model of switching.

2. There are learning algorithms that can construct or refine Bayesian Network structures. This feature is not utilized in our model as the structure of our model is fixed by the logic and the feedback.

3. The temporal links model higher order temporal dependencies, while the non-temporal links models higher order spatial dependencies.

4. These are powerful predictive models and hence works extremely well for estimation. The uniqueness of Bayesian Network based modeling is that evidence can be provided for any node not necessarily just at the inputs. In fact, this feature is used in the probabilistic inference schemes to make the results input pattern insensitive.

5. The inference algorithms can be parallelized, however, we do not exploit this in this paper.

We consider two inference algorithms to form the estimates from the built TC-LiDAG representations: one is an exact scheme and the other is a hybrid scheme. The exact inferences scheme, which is based on local message passing, is presently practical for small circuits due to computational demands on memory. For large circuits, we resort to a hybrid inference method based on combination local message passing and importance sampling. Note that this sampling based probabilistic inference is non-simulative and is different from samplings that are commonly used in circuit simulations. In the later, the input space is sampled, whereas in our case both the input and the line state spaces are sampled simultaneously, using a strong correlative model, as captured by the Bayesian network. Due to this, convergence is faster and the inference strategy is input pattern insensitive.

1.1 Contribution of the Thesis

1. This is the first and only probabilistic modeling framework that can be used to model switching in *both* sequential and combinational circuits. *This is the only completely probabilistic model proposed for sequential circuits.*

2. In this thesis, we theoretically prove that the underlying dependency model of switching activity is a dynamic Bayesian Network as shown in Figure 3.4.c. *The adopted strategy is not logic unraveling.* Switching in every node is inherently connected with the switching at the next time instant. If one were to use just unraveling, then the connection between time slices should really look like Figure 3.4.b. Whereas, the dynamic Bayesian network, being a minimal representation, needs fewer connections between time slices as shown in Figure 3.4.c.

3. The most significant contribution of this thesis is modeling dependencies in the inputs. This is crucial as consecutive input patterns are actually random, however, while modeling switching, they would implicitly be dependent and quantifying this dependence was not trivial. Apart from

that, we capture state dependencies between two consecutive time and quantify them. These two classes of dependencies are responsible and crucial for capturing accurate higher order temporal correlations between time steps and significantly affect the ability to model sequential circuit switching.

4. Last but not the least, We use a non-partition based stochastic inference scheme that (1) scales very well and (2) can result in anytime estimates. Ramani *et al.* [47] recently introduced these stochastic inference schemes to switching estimation that results in almost zero-error estimates in combinational circuits. We use this inference for the dynamic Bayesian Networks. Partition-based inference proposed by Bhanja *et al.* [3, 2] degenerates

   as the error in one time slice is fed back into the next one and propagates through the entire system.

1.2 Organization

We discuss relevant issues and research work in Chapter 2. This chapter deals mostly with statistical simulation-based estimation techniques as our work is the first completely probabilistic approach. Next in Chapter 3, we discuss about the fundamentals of Bayesian Networks and the modeling of a combinational circuit into a Bayesian network. Then we sketch the fundamentals of dynamic Bayesian Networks and also describe the time-coupled-LiDAG model for sequential circuits. We depict probabilistic inference in Chapter 4 and present experimental results in Chapter 5. We conclude the paper in Chapter 6.

CHAPTER 2

PRIOR WORK

Among different types of VLSI circuits, switching in sequential circuits, which also happens to be the most common type of logic, are the hardest to estimate. There are many methods that handle combinational circuit [11, 13, 8, 34, 37, 31, 32, 30, 1, 2, 33] by simulative and probabilistic methods. Many of them cannot be directly applied to sequential circuits. This is particularly due to the complex higher order dependencies in the switching profile, induced by the spatio-temporal components of the main circuit but mainly caused by the state feedbacks that are present. These state feedbacks are not present in pure combinational circuits. One important aspect of switching dependencies in sequential circuits that one can exploit is the first order Markov property, i.e. the system state is independent of all past states given just the previous state. This is true because the dependencies are ultimately created due to logic and re-convergence, using just the current and last values.

The complexity of switching in sequential circuits arise due to the presence of feedback in basic components such as flip-flops and latches. The inputs to a sequential circuit are not only the primary inputs but also these feedback signals. The feedback lines can be looked upon as determining the state of the circuits at each time instant. The state probabilities affect the state feedback line probabilities that, in turn, affect the switching probabilities in the entire circuit. Thus, formally, given a set of inputs i_t at a clock pulse and present states s_t, the next state signal s_{t+1} is uniquely determined as a function of i_t and s_t. At the next clock pulse, we have a new set of inputs i_{t+1} along with state s_{t+1} as an input to the circuit to obtain the next state signal s_{t+2}, and so on. Hence, the statistics of *both* spatial and temporal correlations at the state lines are of great interest. It is important to be able to model both kinds of dependencies in these lines.

Existing techniques for switching estimation in sequential circuits use input pattern simulation. Pure simulation [21] though accurate are expensive in terms of computational time and are strongly pattern dependent. Almost all the statistical techniques in one way or the other has employed sequential

sampling of inputs along with a stopping criteria determined by the assumed statistical model. These methods are weakly pattern dependent, and requires special attention for modeling the mutli-modal power profile. Moreover success of these methods rely greatly on the knowledge of exact input trace either given by the user or generated by a probabilistic model.

In this chapter, we will discuss a few dominant methods for power estimation and briefly go over the only probabilistic modeling effort by Ghosh *et al.* [13]. Since state line statistics are of great concern, Tsui *et al.* [14] presented an exact method using Chapman-Kolmogrov method to estimate switching activity in sequential circuit state line and compared it with an approximate method which was based on calculating the state line probabilities by solving a set of nonlinear equations. These equations were derived from the next state logic. By doing this they were estimating the state line probabilities instead of state probabilities, thereby reducing the complexity of the problem. But the adverse effect of this method is the inability to model the spatial dependencies among the state lines. The set of nonlinear equations, given in Equation. 2.1, were obtained by considering the steady state probabilities for state lines and assuming that input probabilities were known.

$$\begin{aligned} y_1 &= p_1 - g_1(p_1, p_2, ..., p_N) = 0 \\ y_2 &= p_2 - g_2(p_1, p_2, ..., p_N) = 0 \\ &..... \\ y_N &= p_N - g_N(p_1, p_2, ..., p_N) = 0 \end{aligned} \tag{2.1}$$

where, p_i's are state line probabilities, g_i's are nonlinear functions of p_i's and N is the number of flipflops.

In general these equations were written as $Y(P) = 0$ and $P = G(P)$. The approximate state line probabilities were computed by solving $Y(P) = 0$ using Newton-Raphson method or by solving $P = G(P)$ using Picard-Peano method.

The accuracy of the estimate was enhanced by unrolling the next state logic by some user defined limit, as shown in Fig. 2.1.. This actually increased the number of variables and equations, thereby increasing the accuracy.

Chen *et al.* [17] presented a statistical technique to obtain upper and lower bounds of average power dissipation in sequential circuits taking into account the signal probabilities and signal activi-

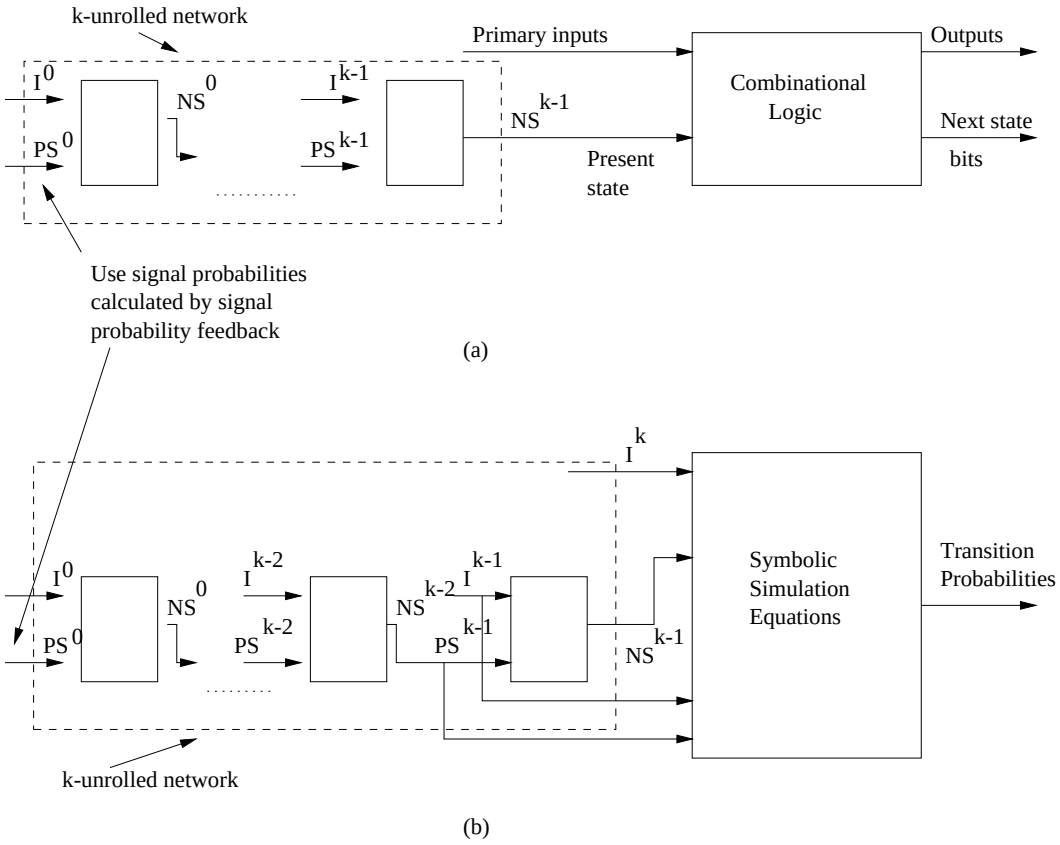


Figure 2.1. Enhancing Accuracy by Unrolling [14].

ties. Through this work the authors emphasized that uncertainties in primary inputs can lead to huge uncertainties in power dissipation. So the basis of this work was to find out power sensitivities due to primary input activities and probabilities. These characteristics were obtained using Equations. 2.2 and 2.3.

$$\zeta a(x_i) = \sum_{j \epsilon allnodes} fanout(j) \frac{\partial a_j}{\partial a(x_i)} \tag{2.2}$$

$$\zeta P(x_i) = \sum_{j \epsilon allnodes} fanout(j) \frac{\partial a_j}{\partial P(x_i)} \tag{2.3}$$

where $\zeta a(x_i)$ is normalized power sensitivity to primary input activity, $\zeta P(x_i)$ is normalized power sensitivity to primary input probability and a_j is normalized activity at node j.

The initial power was obtained using these normalized values. Then the signal properties of the primary inputs were changed one at a time and the circuit was resimulated. Due to the tedious calculations in this method, it is not suitable for large circuits.

Chou *et al.* [18] presented a Monte Carlo based statistical technique to estimate switching activity in sequential circuits. In this work the authors have defined a set of states to be near-closed set if all states inside it have a very small probability to reach any state outside it and vice versa. One such near-closed set was assumed to be the initial state and a warmup simulation period is applied before each sample length period to calculate the probability of the initial near-closed set. In order to obtain a stopping criterion, the number of samples was increased and the sample length was reduced while maintaining the product of the number of samples and sample length as a constant. By doing this they were able to overcome the problem of increased sample deviation when the sample mean was still close to average.

Yuan *et al.* [16] presented a statistical simulation method to estimate average power dissipation in sequential circuits. The flow chart of this method is illustrated in Figure. 2.2.. Since a statistical estimation method needs random samples and the power samples from the sequential circuits are not random, they proposed a method to obtain random power samples. This method included a couple of steps namely a randomness test and the determination of an independence interval. The independence interval was actually used to select samples which are not consecutive (i.e., if independence interval

is 1 then every alternate samples are chosen) thus having a random sequence of samples. At first with an initial independence interval the samples were chosen and they were tested for randomness. The success of the test resulted in acceptance of the independence interval as the suitable one, and the failure results in incrementing the interval and repeating the process. In the randomness test, a particular sequence of samples was considered to be random if it has one or more successive occurrence of identical samples followed or preceded by different samples. Clustering of identical samples or mixing of different samples are considered to be nonrandom sequence of samples.

Murugavel *et al.* [21] proposed a simulation based work for power estimation in both combinational and sequential circuits using Petri Net. They came up with a new form of Petri net called hierarchical colored hardware Petri net (HCHPN).

Najm *et al.* [12] proposed a logic simulation technique to obtain the state line probabilities and further employed a statistical simulation method for power estimation in sequential circuits. The logic simulation was based on the Monte Carlo method and it is employed on the register transfer level (RTL) of the circuit.

Saxena *et al.* [20] enhanced this technique in their work on obtaining state line probabilities using statistical simulation method. Multiple copies of a circuit were simulated using mutually independent input vectors, thereby obtaining mutually independent samples. The simulations were based on Equation. 2.4.

$$\lim_{k\to\infty} P_k(x_i|X_0) = P(x_i) \tag{2.4}$$

where x_i represents a state signal, X_0 is a state at time 0 and $P(x_i)$ is the state line probability. $P_k(x_i|X_0)$ was estimated for increasing values of k until it converges.

Stamoulis [15] proposed a path oriented Monte Carlo method to estimate transition probabilities in sequential circuits. This work was mainly focused on the state lines. The transition probabilities were calculated for each randomly selected path with a randomly selected state as its head while applying random vectors to the primary inputs. The transition probabilities over all these paths were averaged to obtain average switching probabilities over the state lines.

Kozhaya *et al.* [19] formulated a power estimation technique for sequential circuits where a set of input vectors were chosen from a large vector set to obtain upper and lower bounds of power. The input

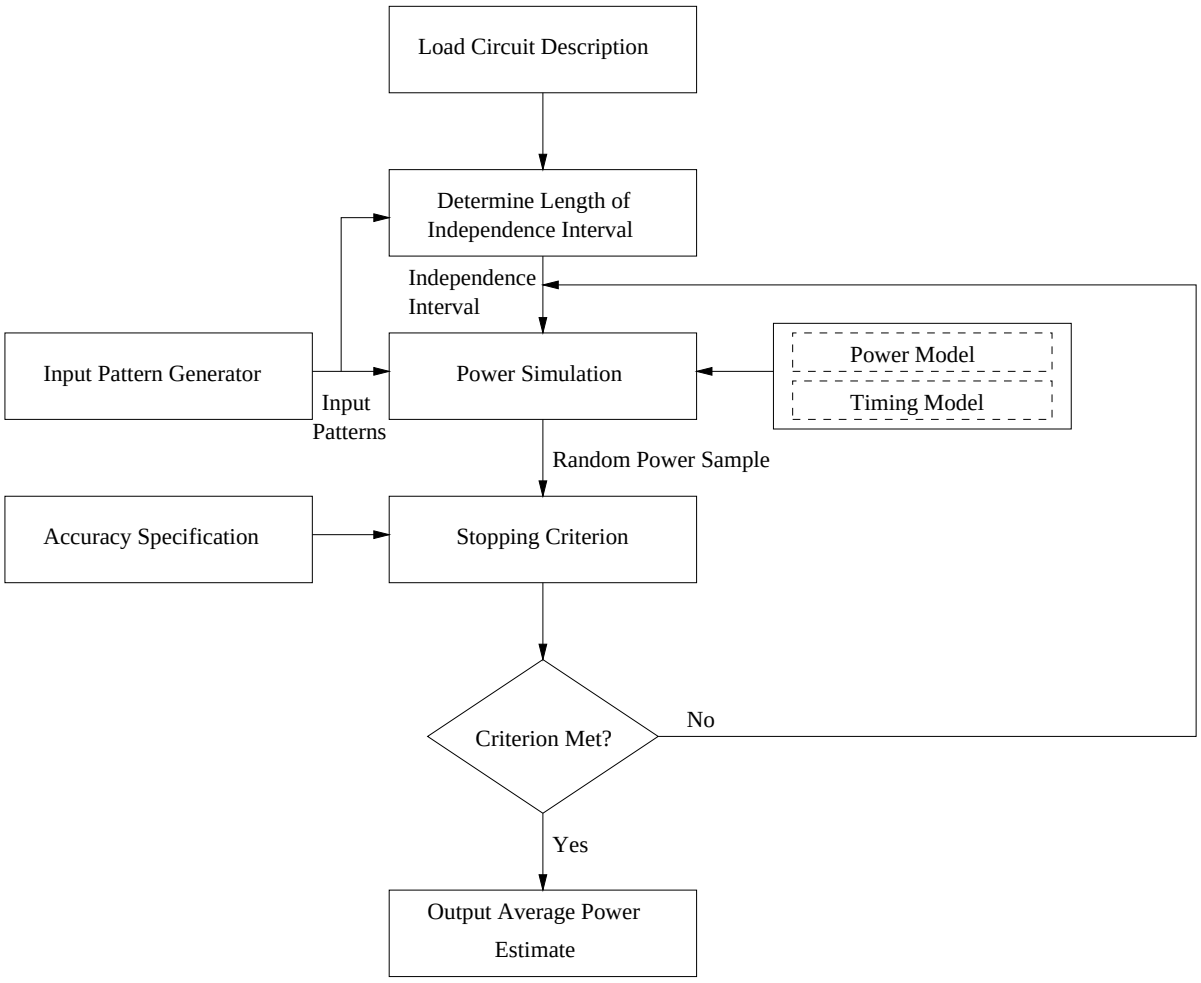


Figure 2.2. Flowchart for the Statistical Estimation of Averae Power Dissipation in Sequential Circuits by Yuan *et al.* [16].

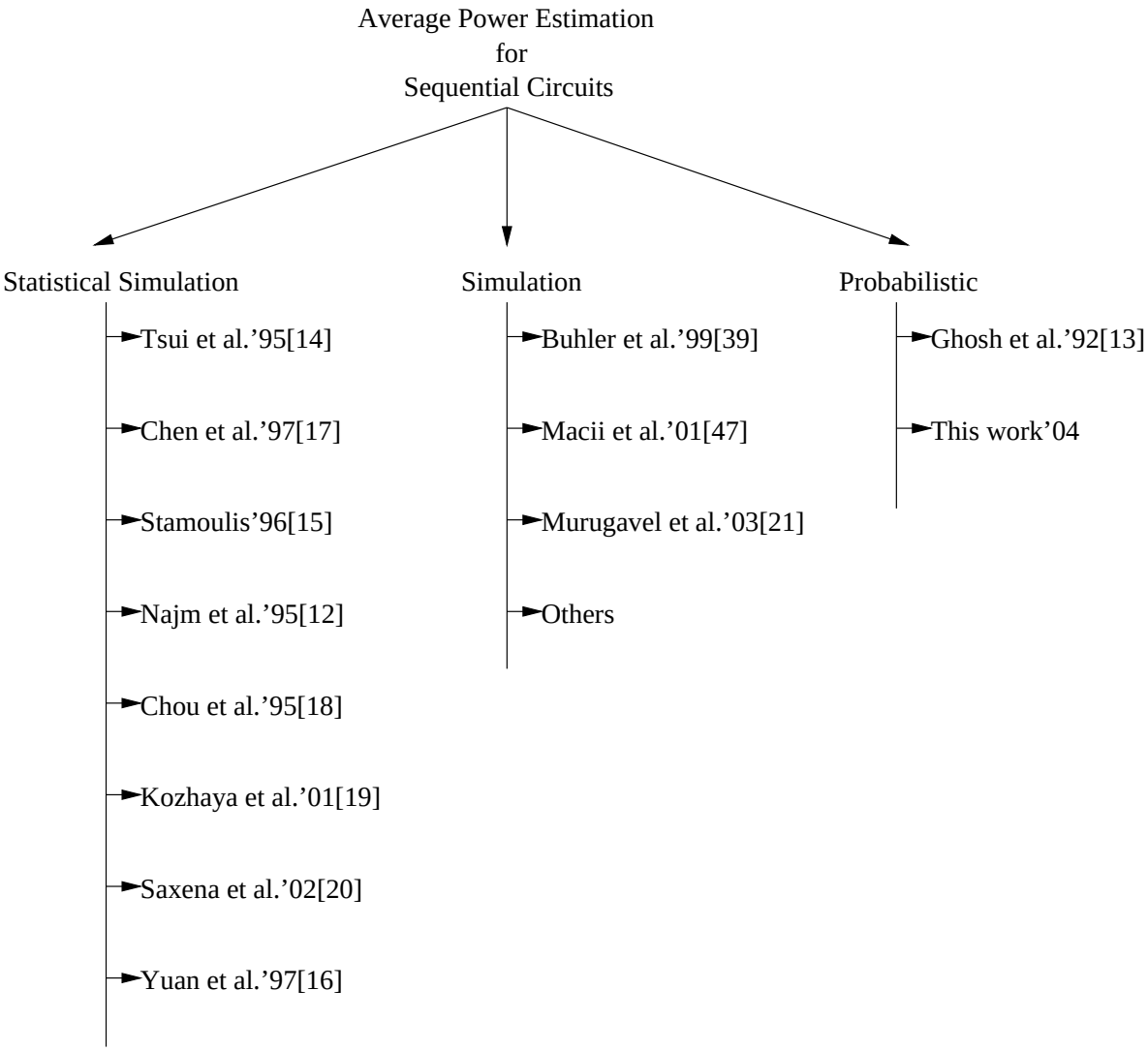


Figure 2.3. Techniques for Estimating Switching Activity in Sequential Circuits.

vectors were separated into blocks and each block was simulated to get corresponding upper and lower bounds of power. A transition from $0 \rightarrow 1$ or from $1 \rightarrow 0$ at the output was accounted for upper bound and a transition from $0 \rightarrow 0$ or from $1 \rightarrow 1$ was accounted for lower bound. Then the upper and lower bound values are calculated using Equations. 2.5 and 2.6.

$$P_K^u(i) = \frac{1}{KT} \sum_{k=i}^{i+K-1} e^u(k) \tag{2.5}$$

$$P_K^l(i) = \frac{1}{KT} \sum_{k=i}^{i+K-1} e^l(k) \tag{2.6}$$

where K is block size, T is clock period and

$$e^u(k) = \frac{1}{2} \sum_j V_{dd}^2 C_j n_k^u(j) \tag{2.7}$$

$$e^l(k) = \frac{1}{2} \sum_j V_{dd}^2 C_j n_k^l(j) \tag{2.8}$$

where C_j is node capacitance, $n_k^u(j)$ and $n_k^l(j)$ are lower and upper bounds on the number of logic transitions made by node j in clock cycle k.

We present the research work done in sequential circuits in the taxonomy diagram shown in Figure 2.3.. As it can be seen most of the previous work, are simulative and hence pattern dependent. Ghosh *et al.* [13] proposed the first attempt for modeling sequential circuits probabilistically, however, they method assumed many independence and hence inaccurate. This work is the first attempt for modeling the sequential circuit probabilistically capturing all correlations both temporal and spatial and with reduced pattern dependence of the estimates.

Bhanja et al [1, 2] proposed Bayesian Network based model for capturing first order temporal and higher order spatial dependence for combinational circuit. This work uses dynamic Bayesian Network based model for capturing the feedback effect. We discuss the fundamentals and probabilistic modeling issues in the next chapter.

CHAPTER 3

DEPENDENCY MODELS FOR SEQUENTIAL CIRCUITS

In this chapter, we develop the dependency model for sequential circuits. Sequential circuits, as we know possesses all the challenges of spatio-temporal dependency modeling that is present in combinational circuits. Moreover, the additional higher order temporal dependencies through the state feedback contributes to directed cycles in the probabilistic model. These higher order temporal dependencies not only make the representation complex but also induces additional spatio-temporal dependencies amongst the present state and the present inputs.

In this chapter, we would first sketch the fundamentals of Bayesian Network and then explain some basic definitions and theorems that are used to structure a Bayesian Network. In the next section, we would present the fundamentals of a Dynamic Bayesian Network followed by discussion about the structure of Dynamic Bayesian Networks. We will conclude this chapter with the modeling and specific issues pertaining to sequential circuit emphasizing the input dependencies and quantification of the temporal edges connecting the inputs in adjacent time slices.

3.1 Bayesian Network Fundamentals

Bayesian Network is a graphical probabilistic model based on the minimal graphical representation of the underlying joint probability function. Due to the probabilistic causal nature, this graphical model is directed and acyclic. On a whole a Bayesian Network can be defined as a directed acyclic graph (DAG) whose nodes are random variables and edges are probabilistic dependencies. The conditional independence that is observed in the probabilistic model is preserved in the graphical structure and moreover none of the edges in the graph can be removed without destroying the conditional independency relationships of the probabilistic model. These features induce a notion of minimal compact representation.

In a combinational circuit, the entire switching profile of the circuit can be represented as a unique joint probability function. The priors to this function are the primary inputs. Representation of combinational circuits as Bayesian Networks has already been proposed by Bhanja et al. in her dissertation works [1], [2], [3].

A Bayesian network is a directed acyclic graph (DAG) representation of the conditional factoring of a joint probability distribution. Any probability function $P(x_1, \cdots, x_n)$ can be written as[1]

$$P(x_1, \cdots, x_N) = P(x_n|x_{n-1}, x_{n-2}, \cdots, x_1)P(x_{n-1}|x_{n-2}, x_{n-3}, \cdots, x_1) \cdots P(x_1) \tag{3.1}$$

This expression holds for any ordering of the random variables. In most applications, a variable is usually not dependent on all other variables. There are lots of conditional independencies embedded among the random variables, which can be used to reorder the random variables and to simplify the conditional probabilities.

$$P(x_1, \cdots, x_N) = \Pi_v P(x_v|Pa(X_v)) \tag{3.2}$$

where $Pa(X_v)$ are the parents of the variable x_v, representing its direct causes. This factoring of the joint probability function can be represented as a directed acyclic graph (DAG), with nodes (V) representing the random variables and directed links (E) from the parents to the children, denoting direct dependencies.

3.2 Conditional Independence Maps

The DAG structure preserves all the *independencies* among sets of random variables and is referred to as a Bayesian network. The concept of Bayesian network can be precisely stated by first defining the notion of *conditional* independence among three *sets* of random variables. The following definitions and theorems appear in [5, 1] and are used later in this paper to prove that the TC-LiDAG structure is a Bayesian network.

Definition 1: Let U=$\{\alpha, \beta, \cdots\}$ be a finite set of variables taking on discrete values. Let $P(.)$ be the joint probability function over the variables in U, and let X, Y and Z be any three subsets (maybe

[1]Probability of the event $X_i = x_i$ will be denoted simply by $P(x_i)$ or by $P(X_i = x_i)$.

overlapping) of U. X and Y is said to be *conditionally independent* given Z if

$$P(x|y,z) = P(x|z) \text{ whenever } P(y,z) > 0 \tag{3.3}$$

Following Pearl [5], we denote this conditional independency amongst X, Y, and Z by $I(X,Z,Y)$; X and Y are said to be *conditionally independent* given Z. A dependency model, M, of a domain should capture all these triplet conditional independencies amongst the variables in that domain. A joint probability density function is one such dependency model. The notion of independence exhibits properties that can be axiomatized by the following theorem [5].

Theorem 1: Let X, Y and Z be three distinct subset of U. If $I(X,Z,Y)$ stands for the relation "X is independent of Y given Z" in some probabilistic model P, then I must satisfy the following four independent conditions:

$$I(X,Z,Y) \Rightarrow I(Y,Z,X) \tag{3.4}$$

$$I(X,Z,Y \cup W) \Rightarrow I(X,Z,Y)\&(X,Z,W) \tag{3.5}$$

$$I(X,Z,Y \cup W) \Rightarrow I(X,Z \cup W,Y) \tag{3.6}$$

$$I(X,Z,Y)\&I(X,Z \cup Y,W) \Rightarrow I(X,Z,Y \cup W) \tag{3.7}$$

Next, we introduce the concept of *d-separation* of variables in a directed acyclic graph structure (DAG), which is the underlying structure of a Bayesian network. This notion of *d-separation* is then related to the notion of independence amongst triple subsets of a domain.

Definition 2: If X, Y and Z are three distinct node subsets in a DAG D, then X is said to be *d-separated* from Y by Z, $< X|Z|Y >$, if there is no path between any node in X and any node in Y along which the following two conditions hold: (1) every node on the path with converging arrows is in Z or has a descendent in Z and (2) every other node is outside Z.

Definition 3: A DAG D is said to be an I-map of a dependency model M if every *d-separation condition* displayed in D corresponds to a valid conditional independence relationship in M, i.e., if for every three disjoint set of nodes X, Y and Z we have, $< X|Z|Y > \Rightarrow I(X,Z,Y)$.

Definition 4: A DAG is a *minimal* I-map of M if none of its edges can be deleted without destroying its dependency model M.

Note that every joint probability distribution function P over a set of variables represents a dependency model M, since it captures all the conditional independencies.

Definition 5: Given a probability distribution P on a set of variable U, a DAG D is called a *Bayesian Network* of P if D is a minimum I-map of P.

There is an elegant method of inferring the minimal I-map of P that is based on the notion of a Markov blanket and a boundary DAG, which are defined below.

Definition 6: A Markov blanket of element $X_i \in U$ is an subset S of U for which $I(X_i, S, U - S - X_i)$ and $X_i \notin S$. A set is called a Markov *boundary,* B_i of X_i if it is a minimal Markov blanket of X_i, i.e. none of its proper subsets satisfy the triplet independence relation.

Definition 7: Let M be a dependency model defined on a set $U = \{X_1, \cdots, X_n\}$ of elements, and let d be an ordering $\{X_{d1}, X_{d2}, \cdots\}$ of the elements of U. The *boundary strata* of M relative to d is an ordered set of subsets of U, $\{B_{d1}, B_{d2}, \cdots\}$ such that each B_i is a Markov boundary (defined above) of X_{di} with respect to the set $U_i(\subset U) = \{X_{d1}, X_{d2}, \cdots, X_{d(i-1)}\}$, i.e. B_i is the minimal set satisfying $B_i \subset U$ and $I(X_{di}, B_i, U_i - B_i)$. *The DAG created by designating each B_i as the parents of the corresponding vertex X_i is called a* boundary DAG *of M relative to d.*

This leads us to the final theorem that relates the Bayesian network to I-maps, which has been proven in [5]. This theorem is the

key to constructing a Bayesian network over multiple time slices (Dynamic Bayesian Networks).

Theorem 2: Let M be any dependency model satisfying the axioms of independence listed in Eqs. 3.4- 3.7. If graph structure D is a boundary DAG of M relative to ordering d, then D is a minimal I-map of M.

3.3 Combinational Circuit as a Bayesian Network

Figure. 3.1.(a) represents a simple combinational circuit and the DAG in Figure. 3.1.(b) illustrates its Bayesian Network representation. The conditions for a DAG structure to be a Bayesian Network are as follows,

1. the presence of conditionally independent set of random variables in the probabilistic model P.

2. the presence of d-seperation of random variables in the DAG D.

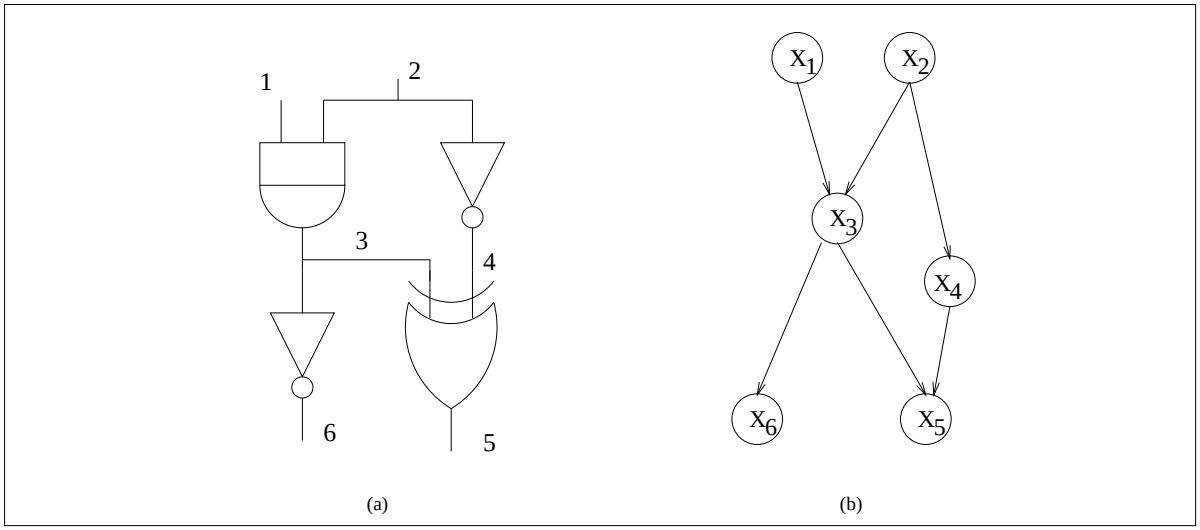


Figure 3.1. A Simple Combinational Circuit.

3. the DAG D should be the minimal I-map of the probability function P.

The following discussion is based on the definitions and theorems presented in Section. 3.2.

In the switching probabilistic model, X_5 is conditionally independent of X_2 given X_3 and X_4. In the DAG structure illustrated in Figure. 3.1.(b), from the conditions of d-seperation, we can clearly point out that X_5 is d-seperated from X_2 by $X_3,\ X_4$.

The method of obtaining minimal I-map of the probabilistic function, say P is based on Markov blanket and a boundary DAG. In the DAG structure illustrated in Figure. 3.1.(b), considering the random variable X_5, S=X_3,X_4 is a Markov blanket, since given X_3,X_4, X_5 is independent of the rest of the random variables in the domain. In the DAG structure illustrated in Figure. 3.1.(b), the boundary strata of underlying dependency model over the domain is given by,

$$B_M = \{\{X_1, X_2\}, \{X_3, X_4\}, \{X_2\}\} \tag{3.8}$$

This clearly depicts that the DAG in Figure. 3.1.(b) is a boundary DAG.

Using these characteristics it has already been proved [5] that, if a graph structure is a boundary DAG of the dependency model, then the graph structure is a minimal I-map of the dependency model. These aspects collectively prove that the DAG structure in Figure. 3.1.(b) is a Bayesian Network.

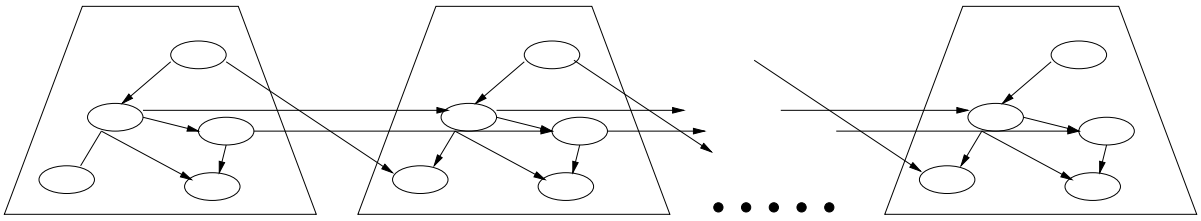


Figure 3.2. Time Slice Model with Snapshot of the Evolving Temporal Process [26].

3.4 Dynamic Bayesian Network

The focus of this work is towards the realization of sequential circuits as a graphical probabilistic model. Due to the feedback process in sequential circuits they cannot be represented as a directed acyclic graph. So eventually they cannot be modeled as a Bayesian Network. Hence we model sequential circuit as time coupled Bayesian Networks which are also called dynamic Bayesian Networks.

Not only the digital circuits but most of the common processes need to be observed at each point of time and possess different probabilistic dependencies at different instant of time. Such dynamic activities are normally analyzed at each time instant and a final output is obtained. The basic Bayesian Network was not designed to handle such dynamic activities. In the beginning the researchers, when they started formulating BN in a new direction, they had to resort to two identical yet different models namely, temporal and dynamic. The temporal model was stated to be a subset of dynamic model, because it cares only about the change in time and not about the change in state. If a system remains in the same state at different time instants, then it is a temporal model. If there is change in state along with time, then it is a dynamic model.

In the discussion about structuring a Dynamic Bayesian Network, one might resort to two different approaches as shown in Figures. 3.2. and 3.3..

Figure. 3.2. represents a time sliced model where each time slice consists of a sub-model representing the belief network at a particular point of time or time interval. The adjacent time slices are temporally connected. Figure 3.3. represents a temporal model where the belief network is duplicated into identical sub-models over each time slice however the state variables are not allowed to be dependent in one time slice. Hence, the structure of Figure 3.3. is not ideal for sequential circuits. The advent of the idea of time slices has formed the basis of structuring a sequential circuit into a BN.

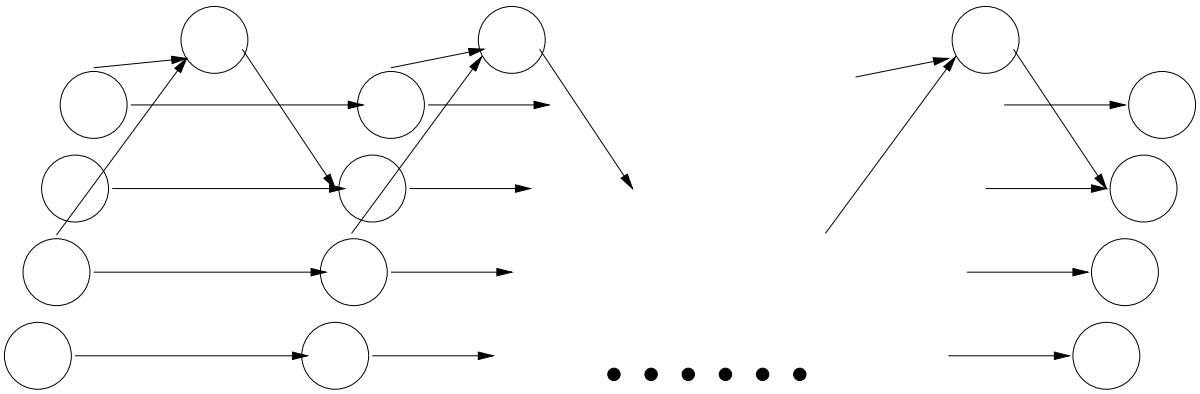


Figure 3.3. Temporal Model with Duplicated Time Slices [26].

As discussed before, Dynamic Bayesian Network (DBN) is a generalization of Bayesian networks to handle temporal effects of an evolving set of random variables. Other formalisms such as hidden Markov models and linear dynamic systems are special cases. The nodes and the links of the DBN are defined as follows. For any time period or slice, t_i, let a directed acyclic graph (DAG), $G_{t_i} = (V_{t_i}, E_{t_i})$, represent the underlying dependency graphical model for the combinational part. Then the nodes of the DBN, V, is the union of all the nodes each time slice.

$$V = \bigcup_{i=1}^{n} V_{t_i} \tag{3.9}$$

However, the links, E, of the DBN are not just the union of the links in the time-slice DAGs, but also include links between time-slices, i.e. temporal edges, $E_{t_i,t_{i+1}}$, defined as

$$E_{t_i,t_{i+1}} = \{(X_{i,t_i}, X_{j,t_{i+1}}) | X_{i,t_i} \in V_{t_i}, X_{j,t_{i+1}} \in V_{t_{i+1}}\} \tag{3.10}$$

where X_{j,t_k} is the j-th node of the DAG for time slice t_k. Thus the complete set of edges E is

$$E = E_{t_1} \cup \bigcup_{i=2}^{n} (E(t_i) + E_{t_{i-1},t_i}) \tag{3.11}$$

Apart from the independencies among the variables from one time slice, we also have the following independence map over variable across time slices if we assume that the random variables representing

the nodes follow Markov property, which is true for switching.

$$I\left(\{X_{j,t_1},\cdots,X_{j,t_{i-1}}\},X_{j,t_i},\{X_{j,t_{i+1}},\cdots,X_{i,t_{i+k}}\}\right) \text{ is true } \forall i>1,k>1 \tag{3.12}$$

3.5 Modeling Sequential Circuit

As we have discussed before, dependency modeling of the underlying switching model for sequential circuits can only be performed by a dynamic Bayesian Networks due the feedback. In this section, we will discuss specific issues with modeling the Markov Blanket of individual variables and quantification of dependencies.

The core idea is to express the switching activity of a circuit as a joint probability function, which can be mapped one-to-one onto a Bayesian Network, while preserving the dependencies. To model switching at a line, we use a random variable, X, with four possible states indicating the transitions from $\{x_{00},x_{01},x_{10},x_{11}\}$. For combinational circuits, directed edges are drawn from the random variables representing switching of each gate input to the random variable for switching at the outputs of that gate. At each node, we also have conditional probabilities, given the states of parent nodes. If the DAG structure follows the logic structure, i.e. we have a logically induced DAG (LiDAG), then it is guaranteed to map all the dependencies inherent in the *combinational* circuit. However, sequential circuits cannot be handled in this manner.

3.5.1 Structure

Let us consider graph structure of a small sequential circuit shown in Fig. 3.4.(a). Following logic structure will not result in a DAG; there will be directed cycles due to feedback lines. To handle this, we do not represent the switching at a line as a single random variable, X_k, but rather as a set of random variables, representing the switching at consecutive time instants, $\{X_{k,t_1},\cdots,X_{k,t_n}\}$, and then model the logical dependencies between them by two types of directed links.

1. For any time instant, edges are constructed between nodes that are logically connected in the combinational part of the circuit, i.e. without the feedback component. Edges are drawn from each random variable representing switching activity at each input of a gate to the random vari-

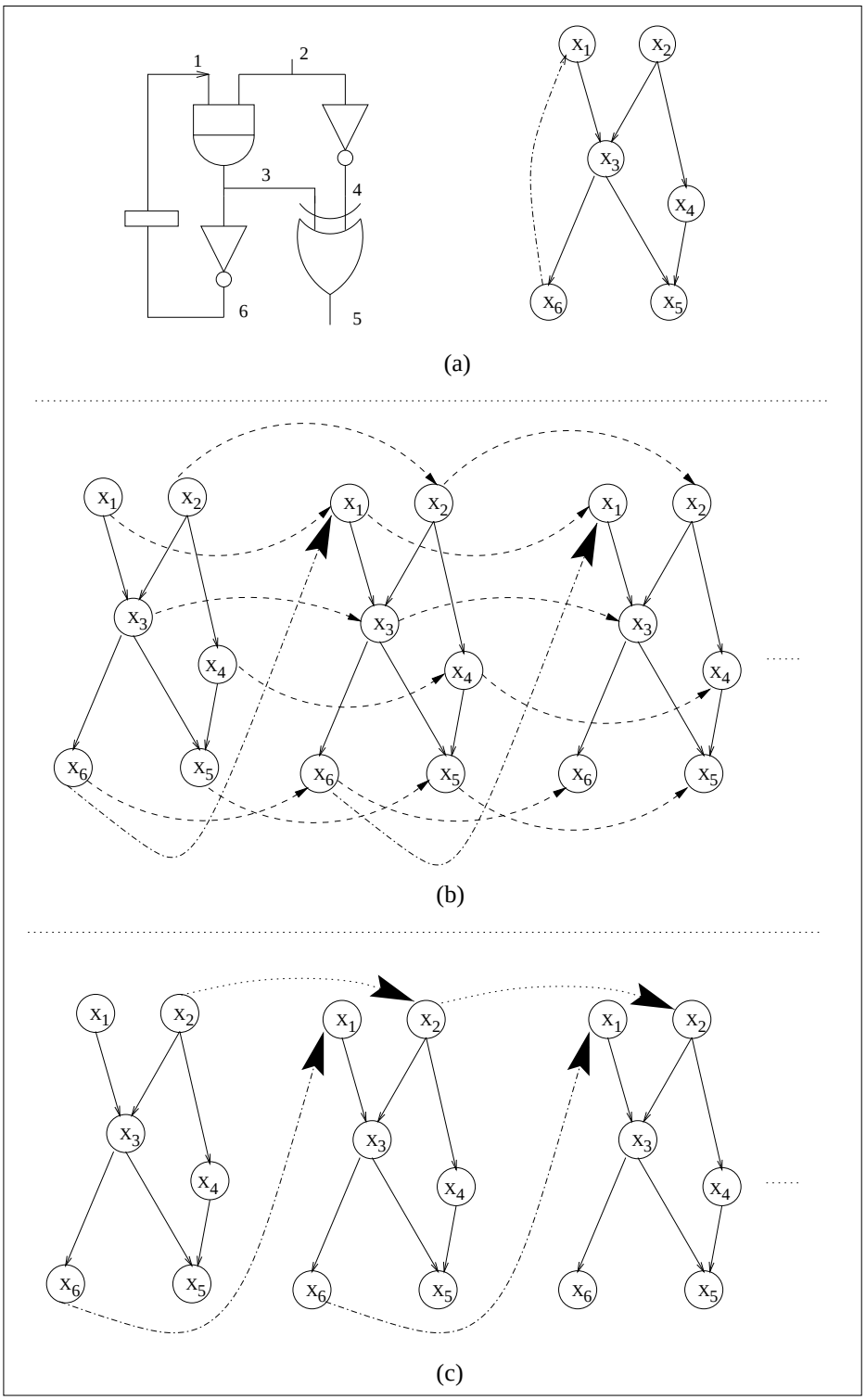


Figure 3.4. (a) A Simple Sequential Circuit and its Graphical Model. (b) Time Unraveled Representation. (c) TC-LiDAG Representation.

able representing output switching of the gate. This gives us the LiDAG structure, capturing the non-sequential nature.

2. We connect random variables representing the same state line from two consecutive time instants, $X^s_{k,t_i} \rightarrow X^s_{k,t_{i+1}}$, to capture the temporal dependencies between the switchings at state lines. Moreover, we also connect the random variables representing the switching at primary input lines at consecutive times, $X^p_{k,t_i} \rightarrow X^p_{k,t_{i+1}}$. This is done to capture the constraint in the primary line switching between two consecutive time instants. For instance, if an input has switched from $0 \rightarrow 1$ at time t_i, then switching at the next time instant cannot be $0 \rightarrow 0$.

We call this graph structure as the time coupled, logically induced DAG or TC-LiDAG. Fig. 3.4.(b) shows the TC-LiDAG for the example sequential circuit in Fig. 3.4. (a); we just show two time slices here. The dash-dot edges shows the second type of edges mentioned above, which couples adjacent LiDAGs. We have X_2 as input and X_1 as the present state node. Random variable X_6 represents the next state signal. Note that this graph is a DAG. We next prove that this TC-LiDAG structure is a minimal representation, hence is a dynamic Bayesian network.

Theorem 3: The TC-LiDAG structure, corresponding to the sequential circuit is a minimal I-map of the underlying switching dependency model and hence is a dynamic Bayesian network.

Proof: Let us order the random variables $\{X_{i,t_i}\}$, such that (i) for two random variables from one time t_i, X_{p,t_i} and X_{c,t_i}, where p is an input line to a gate and c is a output line to the same gate, X_{p,t_i}, appears before X_{c,t_i} in this ordering and (ii) the random variables for the next time slice $t(i+1)$, $\{X_{1,t_{i+1}}, \cdots, X_{n,t_{i+1}}\}$ appear after the random variables at time slice t_i.

With respect to this ordering, the Markov boundary of a node, X_{i,t_i}, is given as follows. If X^p_{i,t_i} represents switching of an input signal line, then its Markov boundary is the variable representing the same input in time slice $X^p_{i,t_{i-1}}$. If X^s_{i,t_i} represents switching of a state signal, then its Markov boundary is the variable representing the switching at the previous time slice $X^s_{i,t_{i-1}}$. And, since the switching of any gate output line is just dependent on the inputs of that gate, the Markov boundary of a variable representing any gate output line consists of just those that represent the inputs to that gate. In the TC-LiDAG structure the parents of each node are its Markov boundary elements hence the TC-LiDAG is a boundary DAG. And, by Theorem 2 the TC-LiDAG is a minimal I-map and thus a Bayesian network

(BN). Since nodes and the edges in the TC-LiDAG defined over n time slices can be described by Equation. 3.9, and Equation. 3.11, the TC-LiDAG is a dynamic Bayesian Network (DBN).

3.5.2 Spatial Dependency Quantification

In TC-LiDAG, we have the random variables that represent switching at each signal at a time instant t which indicates the transitions from $0 \to 0, 0 \to 1, 1 \to 0, 1 \to 1$.

In this section we will discuss the spatial edges i.e. the dependencies that arise between signals in one instant of time. Directed edges are drawn from the random variables representing switching of the inputs to the random variable for switching at the output of each gate. Note that these dependencies denote the spatial correlations amongst the variables in one time instant. The conditional probability of random variable representing switching at an output variable given its parents are determined purely from the logical structures.

The conditional probabilities of the lines that are directly connected by a gate can be obtained knowing the type of the gate. For example, $P(X_3 = x_{01}|X_1 = x_{01}, X_2 = x_{00})$ will be always 0 because if one of the inputs of an AND gate makes a transition from 0 to 1 and the other stays at 0 then the output does not change and hence $P(X_3 = x_{01}|X_1 = x_{01}, X_2 = x_{00}) = 0$. A complete specification of the conditional probability of $P(x_3|x_1, x_2)$ will have 4^3 entries since each variable has 4 states. These conditional probability specifications are determined by the gate type. Thus, for an AND gate, if one input switches from 0 to 1 and the other from 1 to 0, the output remains at 0. We describe the conditional probability specification for a two input AND gate in Table 3.1. By specifying a detailed conditional probability we ensure that the spatio-temporal effect (first order temporal and higher order spatial) of any node are effectively modeled.

It has to be noted that in a single Bayesian Network, the temporal dependencies are in general not modeled. For combinational circuit, Bhanja *et al.* [2] used the states as four possible temporal transitions of a signal $0 \to 0$, $0 \to 1$, $1 \to 0$, $1 \to 1$ to model first order temporal dependence which is sufficient for modeling zero-delay combinational circuits. In next subsection, we will discuss the modeling issues pertaining to the higher order temporal dependencies.

Table 3.1. Conditional Probability Specifications for the Output and the Input Line Transitions for Two Input AND Gate [2].

Two Input AND gate					
$P(X_{output}\|X_{input1}, X_{input2})$ for $X_{output} =$ $\{x_{00}$	x_{01}	x_{10}	$x_{11}\}$	X_{input1} =	X_{input2} =
1	0	0	0	x_{00}	x_{00}
1	0	0	0	x_{00}	x_{01}
1	0	0	0	x_{00}	x_{10}
1	0	0	0	x_{00}	x_{11}
1	0	0	0	x_{01}	x_{00}
0	1	0	0	x_{01}	x_{01}
1	0	0	0	x_{01}	x_{10}
0	1	0	0	x_{01}	x_{11}
1	0	0	0	x_{10}	x_{00}
1	0	0	0	x_{10}	x_{01}
0	0	1	0	x_{10}	x_{10}
0	0	1	0	x_{10}	x_{11}
1	0	0	0	x_{11}	x_{00}
0	1	0	0	x_{11}	x_{01}
0	0	1	0	x_{11}	x_{10}
0	0	0	1	x_{11}	x_{11}

Table 3.2. Conditional Probability Specification between State Line Switchings at Consecutive Time Instants: $P(x^s_{k,t_{i+1}}|x^s_{k,t_i})$.

$X^s_{k,t_{i+1}} =$				X^s_{k,t_i}
$\{x_{00}$	x_{01}	x_{10}	$x_{11}\} =$	
1	0	0	0	x_{00}
0	1	0	0	x_{01}
0	0	1	0	x_{10}
0	0	0	1	x_{11}

Table 3.3. General Example of Switching of a Signal X_k representing the Value of the Signal at each Time Instant t.

Time	Input	Switching	Input	Switching
t_i	I_1	X_1	I_2	X_2
0	0	$0 \rightarrow 0$	1	$0 \rightarrow 1$
1	1	$0 \rightarrow 1$	1	$1 \rightarrow 1$
2	1	$1 \rightarrow 1$	1	$1 \rightarrow 1$
3	0	$1 \rightarrow 0$	1	$1 \rightarrow 1$
4	1	$0 \rightarrow 1$	0	$1 \rightarrow 0$
5	0	$1 \rightarrow 0$	1	$0 \rightarrow 1$
6	0	$0 \rightarrow 0$	0	$1 \rightarrow 0$
7	0	$0 \rightarrow 0$	0	$0 \rightarrow 0$

3.5.3 Temporal Dependency Quantification

The joint probability function is modeled by a Bayesian network as the product of the conditional probabilities defined between a node and its parents in the TC-LiDAG structure: $P(x_v|Pa(X_v))$. These conditional probabilities can be easily specified using the circuit logic. We demonstrate handling the internal lines in the previous subsection. There are two basic types of conditional probability specifications for the temporal edges between (i) primary input lines, and (ii) state lines. For state lines, the conditional probability models the logic of a buffer, as shown in Table 3.2..

Please consider the example of an input pattern in Table 3.3.. Each row of the table indicate one time instant, column 2 and 4 show the actual values of the signals and column 3 and 5 show the corresponding switchings. Note that the inputs are purely independent, however, switchings are not. If switching is $0 \rightarrow 1$ in row 2, it cannot be $0 \rightarrow 0$ in row 3. Hence under purely random input situations, switching of a variable can take only two values (say $0 \rightarrow 1$, $0 \rightarrow 0$) out of four states in (t+1) given one particular value (say $0 \rightarrow 0$) at t.

Table 3.4. Conditional Probability Specification between Primary Input Line Switchings at Consecutive Time Instants: $P(x^p_{k,t_{i+1}}|x^p_{k,t_i})$.

$X^p_{k,t_{i+1}} = \{x_{00}$	x_{01}	x_{10}	$x_{11}\} =$	X^p_{k,t_i}
0.5	0.5	0	0	x_{00}
0	0	0.5	0.5	x_{01}
0.5	0.5	0	0	x_{10}
0	0	0.5	0.5	x_{11}

Hence for primary input lines, the conditional probabilities models the switching constraints between two time instants, as listed in Table 3.4.. For instance, if the primary line switched from 0 to 1, then at the next time slice the line can either switch from 1 to 0 or remain at 1. Since, we are considering random inputs, we distribute the probabilities equally (with 0.5 probability) between the two options. For correlated inputs, these conditional probabilities will have to be adjusted.

CHAPTER 4

PROBABILISTIC INFERENCE

In the previous section, we discussed modeling of sequential circuits as dynamic Bayesian Networks. The attractive feature of this modeling is that the conditional independence relationships not only help in modeling causality but is also an instrument for probabilistic inference. In this section, we first discuss one of the exact inference schemes that we use to validate our modeling. The exact inferences scheme, which is based on local message passing, is presently practical for small circuits due to computational demands on memory. For large circuits, we resort to a hybrid inference method based on combination local message passing and sampling.

4.1 Exact Inference

The exact inference scheme is based on local message passing on a tree structure, whose nodes are subsets (cliques) of random variables in the original DAG [4, 9]. This tree of cliques is obtained from the initial DAG structure via a series of transformations that preserve the represented dependencies. The original DAG is first converted into an undirected Markov graph structure, which is referred to as the *moral graph*, modeling the underlying joint probability distribution. This moral graph is obtained from the DAG structure, by adding undirected links between the parents of a common child node. These additional links directly capture the dependencies that were only implicitly represented in the DAG. In a moral graph, every parent-child set form a complete subgraph. Due to the undirected

nature of the moral graph, some of the *independencies* represented in the DAG would be lost, resulting in a non-minimal representation. The dependency structure is, however, preserved. This loss of minimal representation will eventually result in increased computational demands, but does not sacrifice accuracy.

Next, a chordal graph is obtained from the moral graph by triangulating it. Triangulation is the process of breaking all cycles in the graph to be composition of cycles over just 3 nodes by adding

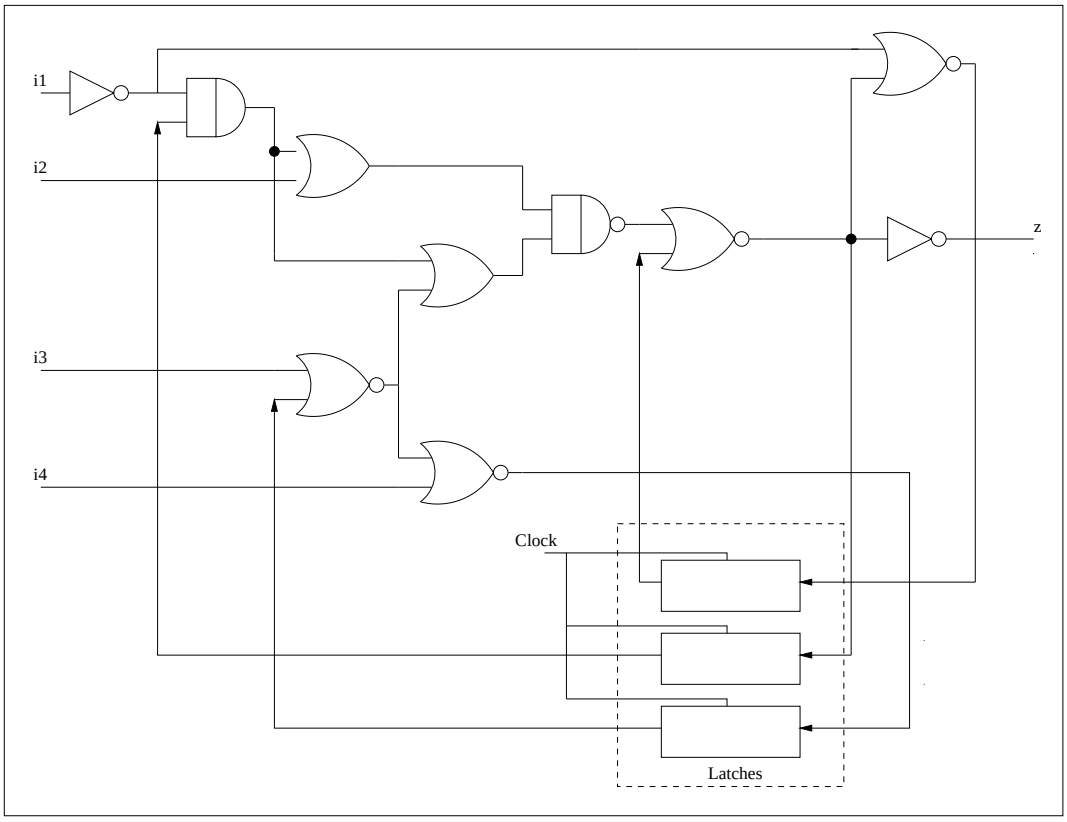


Figure 4.1. Circuit Diagram of *s*27.

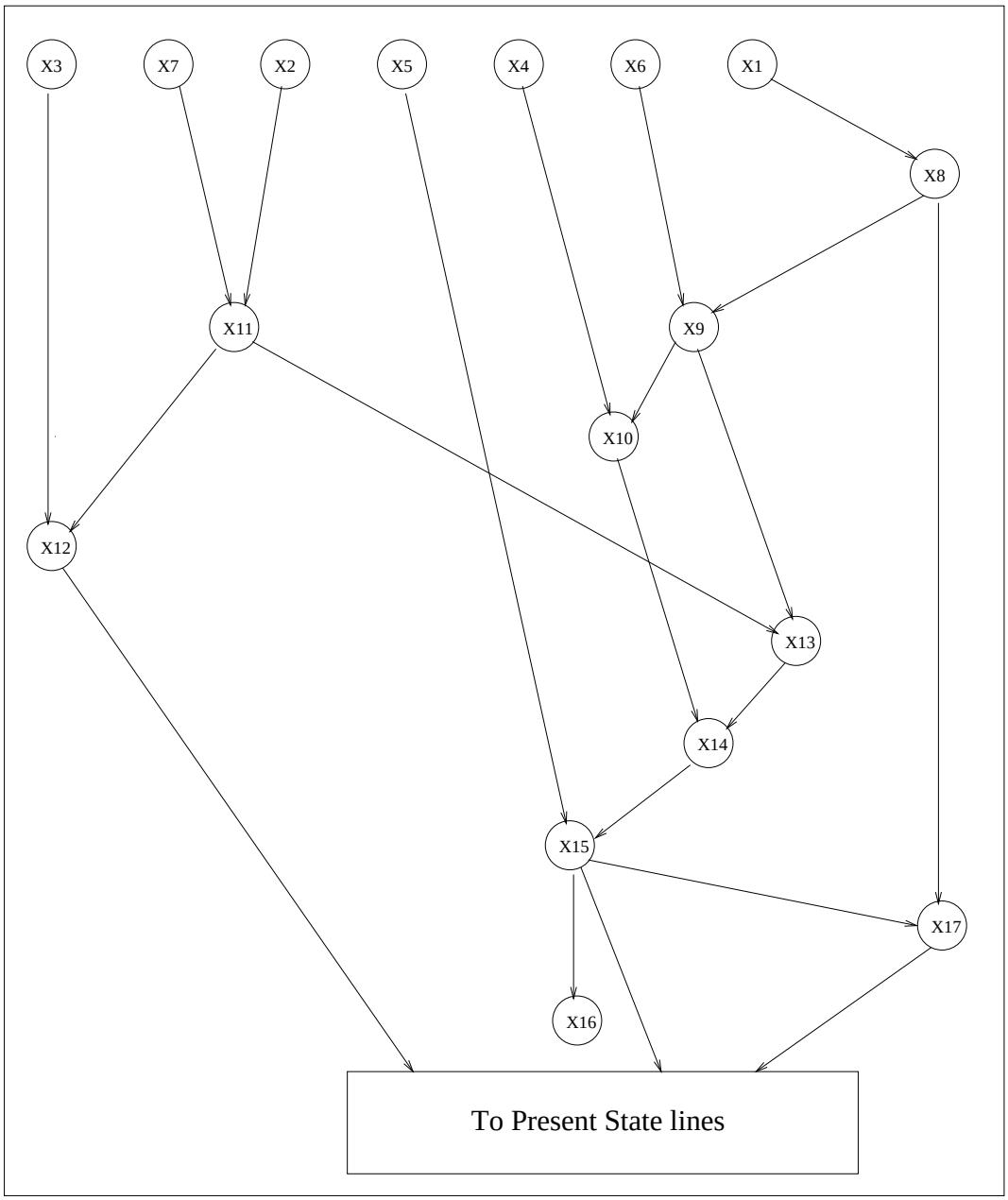


Figure 4.2. Bayesian Network Model of the Combinational Part of *s*27 for One Time Slice.

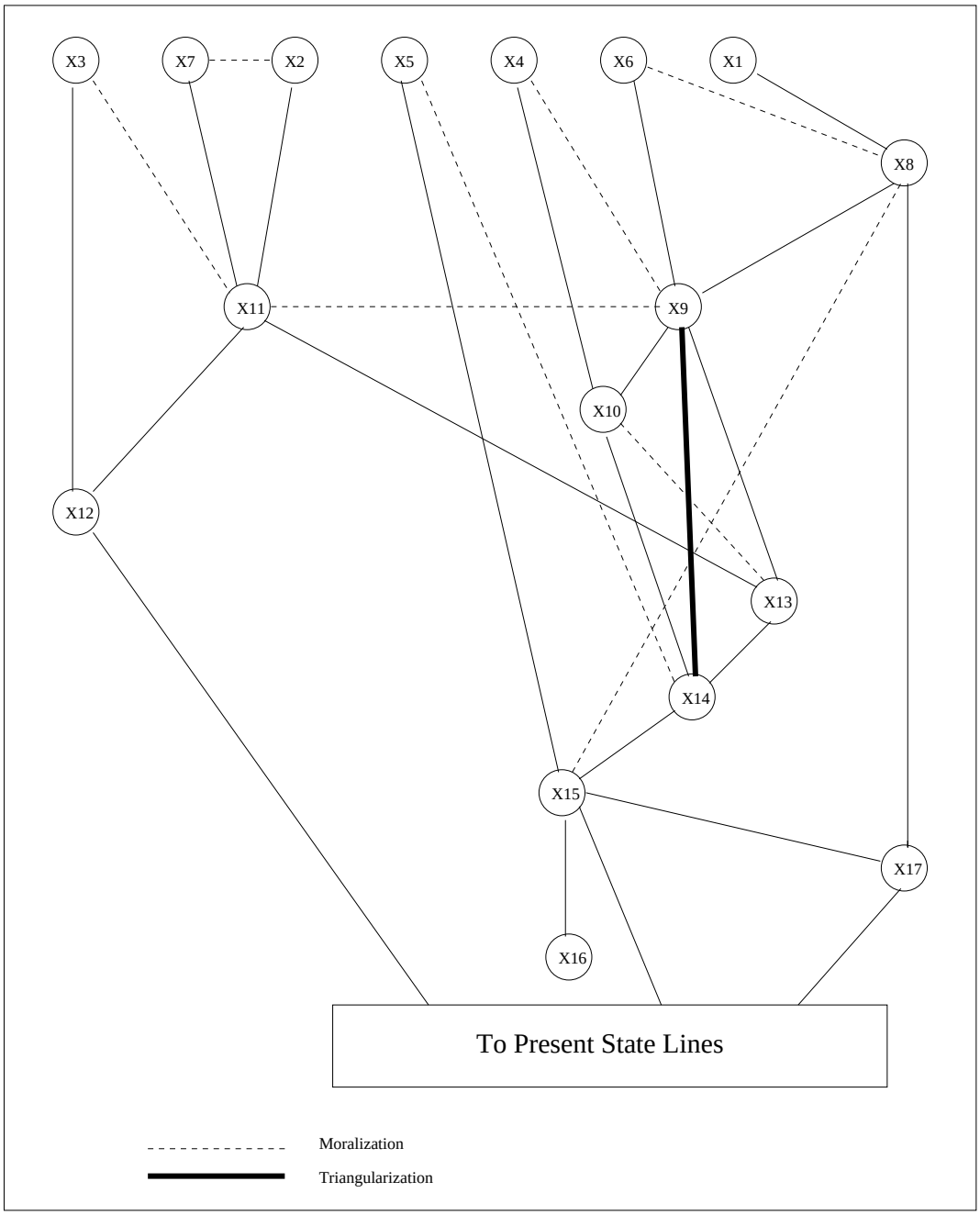
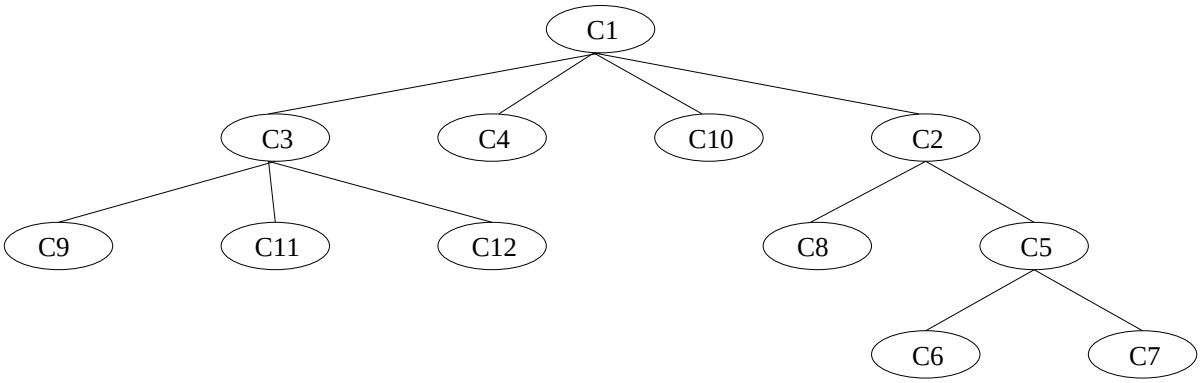


Figure 4.3. Triangulated Undirected Graph Structure of *s*27.



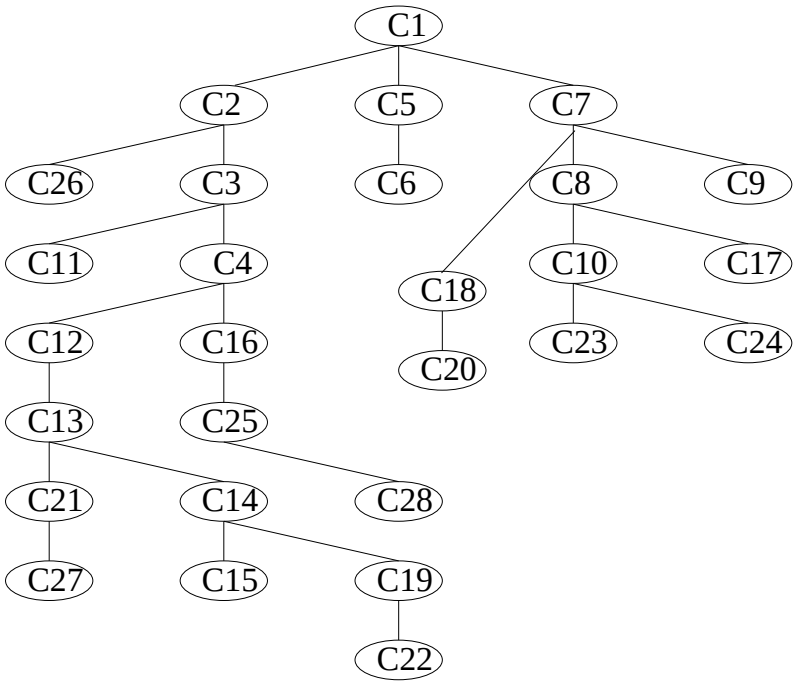
C1 = {X8,X9,X14}
C2 = {X9,X14,X10,X13}
C3 = {X8,X14,X15}
C4 = {X8,X1}
C5 = {X9,X13,X11}
C6 = {X11,X12,X7}
C7 = {X11,X3,X12}
C8 = {X9,X10,X4}
C9 = {X14,X15,X5}
C10 = {X8,X9,X6}
C11 = {X15,X16}
C12 = {X8,X15,X17}

Figure 4.4. Junction Tree of Cliques for One Time Slice Model of *s*27.

additional links. To control the computational demands, the goal is to form a triangulated moral graph with minimum number of additional links. Various heuristics exist for this. For instance, the Bayesian network inference software HUGIN (www.hugin.com), which we use in this work, uses efficient and accurate minimum fill-in heuristics to calculate these additional links.

Cliques of this chordal graph form the nodes of the junction tree. The tree structure is useful for local message passing. Given any evidence, messages consist of the updated probabilities of the common variables between two neighboring cliques. Global consistency is automatically maintained by constructing the tree in such a way that any two cliques, sharing a set of common variables, should have these common variables present in all the cliques that lie in the connecting path between the two cliques. A junction tree with this property can be easily obtained from the same minimum fill-in heuristic algorithm that is used to triangularize the graph [2].

As an example of clique tree construction process, we consider the gate level circuit diagram for *s*27, shown in Fig. 4.1.. The corresponding DAG Bayesian network model, over one time slice, is shown in Fig. 4.2.. We first illustrate the process on this partial model. The inference scheme for the complete dynamic Bayesian network model, as captured by the TC-LiDAG, would be similar. The moralized and triangularized form is shown in Fig. 4.3.. The clique tree is shown in Fig. 4.4.. One interesting feature of this exact modeling is that probabilistic updating is really fast once the compilation is performed, and hence is useful for fast design-space exploration. The probabilities are



C1 = {X4b,X9b,X11b,X14b,X8a,X15a} C8 = {X14b,X8a,X15a,X8b,X5b}
C2 = {X4b,X11b,X8a,X15a,X9a} C9 = {X9b,X15aX8b,X6b}
C3 = {X4b,X11b,X15a,X9a,X8a,X10a} C10 = {X14b,X8b,X5b,X15b}
C4 = {X11b,X15a,X9a,X10a,X13a} C11 = {X4b,X9a,X10a,X4a}
C5 = {X4b,X9b,X11b,X14b,X10b} C12 = {X11b,X9a,X13a,X11a}
C6 = {X9b,X11b,X14b,X10b,X13b} C13 = {X11b,X11a,X2b,X7b}
C7 = {X9b,X14b,X8a,X15a,X8b} C14 = {X11b,X11a,X17b,X3a}
C15 = {X11a,X7b,X3a,X12a} C22 = {X11b,X3b,X12b}
C16 = {X15a,X10a,X13a,X14a} C23 = {X15b,X16b}
C17 = {X8a,X15a,X5b,X17a} C24 = {X8b,X15b,X17b}
C18 = {X8a,X8b,X1b} C25 = {X15a,X14a,X5a}
C19 = {X11b,X3a,X3b} C26 = {X8a,X9a,X6a}
C20 = {X8a,X1b,X1a} C27 = {X11a,X2a,X7a}
C21 = {X11a,X2b,X2a} C28 = {X15a,X16a}

Figure 4.5. Junction Tree of Cliques for Two Time Slices of *s*27.

propagated through the junction tree just by local message-passing between the adjacent cliques. Let us consider two neighboring cliques to understand the key feature of the Bayesian updating scheme. Let two cliques *A* and *B* have probability potentials ϕ_A and ϕ_B, respectively, obtained by multiplying the conditional probabilities, in the DAG based Bayesian network, involving the nodes in each clique. Let *S* be the set of common nodes between cliques *A* and *B*. The two neighboring cliques have to agree on probabilities on the node set *S*, which is termed their separator. To achieve this, we first compute the marginal probability of *S* from probability potential of clique *A* and then use that to scale the probability potential of *B*. The transmission of this scaling factor, which is needed in updating, is referred to as message passing. New evidence is absorbed into the network by passing such local messages. The pattern of the message is such that the process is multi-threadable and partially parallelizable. Because the junction tree has no cycles, messages along each branch can be treated independent of the others.

The final junction tree of cliques for the TC-LiDAG structure of *s*27 for two time slices is more complicated, as shown in Fig. 4.5.. Notice the increased number of cliques. In general, the size of the maximal clique will increase. This result in increased memory requirement to store the probability potential over the nodes in the cliques; the increase is exponential in the maximal clique size. Thus, it is obvious that the exact model cannot be used for large circuits. Available memory would determine the maximum circuit size that can be modeled exactly. In this work, we use this inference only for model validation with small circuits.

4.2 Hybrid Scheme

For large circuits, a hybrid scheme, specifically the pre-propagated importance sampling (EPIS) [22, 23], which uses local message passing and stochastic sampling, is appropriate. This method scales well with circuit size and is proven to converge to correct estimates. These classes of algorithms are also anytime-algorithms since they can be stopped at any point of time to produce estimates. Of course, the accuracy of estimates increases with time. The other useful method is Probabilistic Logic Sampling (PLS) method.

The EPIS algorithm is based on importance sampling that generates sample instantiations of the *whole* DAG network, i.e. all for line switching in our case. These samples are then used to form the final estimates. This sampling is done according to an importance function. In a Bayesian network, the

product of the conditional probability functions at all nodes form the optimal importance function. Let $X = \{X_1, X_2, \cdots, X_m\}$ be the set of variables in a Bayesian network, $Pa(X_k)$ be the parents of X_k, and E be the evidence set. Then, the optimal importance function is given by

$$P(X|E) = \prod_{k=1}^{m} P(x_k|Pa(x_k, E) \tag{4.1}$$

This importance function can be approximated as

$$P(X|E) = \prod_{k=1}^{m} \alpha(Pa(X_k))P(x_k|Pa(X_k))\lambda(X_k) \tag{4.2}$$

where $\alpha(Pa(X_k)) = P(x_k|E^+)$ and $\lambda(X_k) = P(E^-|x_k)$, with E^+ and E^- being the

evidence from above and below, respectively, as defined by the directed link structure. Calculation of λ is computationally expensive and for this, Loopy Belief Propagation (LBP) [27] over the Markov blanket of the node is used. Yuan *et al.* [23] proved that for a poly-tree, the local loopy belief propagation is optimal. The importance function can be further approximated by replacing small probabilities with a specific cutoff value [22].

This stochastic sampling strategy works because in a Bayesian Network the product of the conditional probability functions for all nodes is the optimal importance function. Because of this optimality, the demand on samples is low. We have found that just thousand samples are sufficient to arrive at good estimates for the ISCAS89 benchmark circuits. Note that this sampling based probabilistic inference is non-simulative and is different from samplings that are used in circuit simulations. In the latter, the input space is sampled, whereas in our case both the input and the line state spaces are sampled simultaneously, using a strong correlative model, as captured by the Bayesian network. Due to this, convergence is faster and the inference strategy is input pattern insensitive.

Probabilistic Logic Sampling developed by Henrion in 1988 is credited to be the first stochastic sampling method for inferencing Bayesian Networks. In this method sampling is performed in the forward direction(from parents to children). In the circuit which is represented as a Bayesian network each node is selected in top-down fashion and they are sampled. While inferencing the Bayesian network the samples are grouped into sets and the observed value in each sample in a set is compared with the corresponding evidence values. If they are inconsistent with each other the whole sample set

is discarded. The same method is repeated with each sample set. In the selected sample set the belief distributions are calculated by averaging the frequencies with which the relevant events occur. As compared to the computational merits, this method also has some disadvantages. Since it is based on forward sampling, the evidence that have already occurred cannot be accounted until the corresponding variables are sampled. The occurrence of unlikely evidence can result in rejection of large number of samples thereby hindering the performance of this method.

CHAPTER 5

EXPERIMENTAL RESULTS

We have used the sequential circuits from the ISCAS89 benchmark suite to verify our method. To generate the "ground truth" estimates to compare against the circuits were simulated, with zero gate delay, for 100,000 test vectors. The root nodes of the TC-LiDAG representation of the sequential circuits, which are the state lines of the first time slice and the primary input, need prior probability specifications. The priors for the primary input lines in the first time-slice of DBN were chosen to be equal, i.e. equally probable switching states. A startup simulation with 50 random test vectors was performed and these startup estimates of the present state lines are given to the first time-slice of the DBN. The exact inference on the

TC-LiDAG structure was done using the commercially available HUGIN software. And, we used a tool named "GeNIe" [7] to implement the hybrid inference strategy based on sampling and loopy belief propagation (EPIS). The tests were performed on a Pentium IV, 2.00GHz, Windows XP computer.

First, we show some results that validates the TC-LiDAG model. For this, we use the *s*27 benchmark circuit. Table 5.1. lists the switching estimates at each line in the circuit as computed by (i)

simulation, (ii) the exact inference scheme based on tree of cliques, and (iii) the hybrid EPIS scheme. We used 10 time slices for the TC-LiDAG representation. Note the excellent agreement of the exact inference scheme with simulation, thus validating that the TC-LiDAG is capturing the high order temporal and spatial correlations. The hybrid inference scheme also results in excellent estimates, close to the exact ones.

Second, we present results on rest of the ISCAS'89 circuits in the form of estimation error statistics, as shown in Table 5.2.. We list both the average error, E_{μ}, and maximum error, $E_{\max}$, over all the nodes. We also list the percentage of nodes with switching error above 2 standard deviations from the mean error; this gives an idea about the error distribution. The listed elapsed times are obtained by the $ftime$ command in the WINDOWS environment, and is the sum of CPU, memory access *and* I/O time.

Table 5.1. Switching Probability Estimates at each Line of the *s*27 Benchmark Circuit as Computed by Simulation, by the Exact Scheme, and by the Hybrid EPIS Method.

Nodes	Simulation	Exact (Clique Tree)	Hybrid (EPIS)
1	0.498	0.500	0.512
2	0.598	0.500	0.494
3	0.501	0.500	0.487
4	0.500	0.500	0.508
5	0.450	0.452	0.463
6	0.123	0.123	0.123
7	0.333	0.333	0.368
8	0.498	0.500	0.512
9	0.078	0.078	0.073
10	0.460	0.461	0.476
11	0.333	0.333	0.334
12	0.333	0.333	0.329
13	0.311	0.311	0.311
14	0.229	0.230	0.234
15	0.123	0.123	0.126
16	0.123	0.123	0.126
17	0.450	0.452	0.461

The TC-LiDAG structure, modeling the sequential circuits, used 3 time-slices and just 1000 sampling iterations were used for the hybrid inference scheme. We see that even for larger benchmarks like *s*5378 the mean error is extremely small. The maximum errors for most circuits are also low, except for $s208, s953$ and *s*5378. However, these maximum errors seem to be isolated to a few nodes as is seen from the low fraction of nodes with error above 2σ. In most cases, only 5% of the nodes exceed this error bound, except for *s*208, where we see that % of nodes in $> E_\mu + 2E_\sigma$ range is around 9%. We also found the accuracy of our model is excellent even for larger benchmark circuits like *s*15850 ($E_\mu = 0.004$), but at the expense of computation time (45 minutes).

Third, we study the effect of varying the modeling and inference parameters, i.e the number of time slices modeled by the TC-LiDAG and number of sampling iteration of the hybrid inference scheme. In Table 5.3., we present results based on 3000 sampling iterations. The time slices are still restricted to three. We see that the average of the mean error estimates for ten benchmarks is 0.006. The average time for estimation has, of course, increased, from 6.66 seconds for 1000 samples to 9.48 seconds for 3000 samples. The average maximum error is reduced from 0.086 to 0.083 for the ten benchmarks. Increasing the sampling iterations to 5000, shown in Table 5.4., we get average mean error of 0.005,

Table 5.2. Switching Activity Estimation Error Statistics based on TC-LiDAG Modeling, using 3 Time Slices, and Hybrid Inference Scheme, using 1000 Samples, for ISCAS'89 Benchmark Sequential Circuits.

Circuits	Mean Error (E_μ)	Maximum Error (E_{max})	Time (s) (CPU+I/O)	% of nodes $> E_\mu + 2E_\sigma$
s27	0.015	0.068	0.047	5.88
s208	0.014	0.234	0.719	9.02
s382	0.002	0.096	2.094	4.95
s444	0.003	0.083	2.734	4.88
s526	0.003	0.044	3.922	3.23
s713	0.008	0.043	9.093	4.92
s820	0.002	0.049	10.060	8.33
s953	0.009	0.162	8.343	7.50
s1196	0.001	0.050	15.060	4.81
s1238	0.001	0.038	14.620	5.00
s1423	0.010	0.127	21.120	6.15
s5378	0.002	0.402	378.680	5.08

Table 5.3. Switching Activity Estimation Error Statistics based on TC-LiDAG Modeling, using 3 Time Slices, and Hybrid Inference Scheme, using 3000 Samples, for ISCAS'89 Benchmark Sequential Circuits.

Circuits	Mean Error (E_μ)	Maximum Error (E_{max})	Time (s) (CPU+I/O)	% of nodes $> E_\mu + 2E_\sigma$
s27	0.023	0.085	0.063	5.88
s208	0.015	0.225	1.219	9.02
s382	0.002	0.094	3.625	5.49
s444	0.004	0.052	4.328	4.88
s526	0.003	0.058	6.359	2.31
s713	0.008	0.056	12.984	4.25
s820	0.003	0.047	14.172	5.45
s953	0.010	0.187	12.031	8.18
s1196	0.001	0.019	20.296	6.06
s1238	0.000	0.017	19.766	4.82
s1423	0.007	0.121	27.547	5.35
s5378	0.001	0.393	404.886	5.21

Table 5.4. Switching Activity Estimation Error Statistics based on TC-LiDAG Modeling, using 3 Time Slices, and Hybrid Inference Scheme, using 5000 Samples, for ISCAS'89 Benchmark Sequential Circuits.

Circuits	Mean Error (E_μ)	Maximum Error (E_{max})	Time (s) (CPU+I/O)	% of nodes $> E_\mu + 2E_\sigma$
s27	0.019	0.081	0.078	5.88
s208	0.015	0.211	1.672	9.02
s382	0.001	0.089	5.157	5.49
s444	0.004	0.052	6.110	3.41
s526	0.000	0.048	8.796	2.31
s713	0.006	0.044	16.891	9.62
s820	0.001	0.035	18.282	5.77
s953	0.009	0.171	15.765	7.73
s1196	0.001	0.022	25.546	5.35
s1238	0.000	0.017	25.609	5.55
s1423	0.009	0.131	33.000	5.48
s5378	0.001	0.389	432.908	5.11

Table 5.5. Switching Activity Estimation Error Statistics based on TC-LiDAG Modeling, using 3 and 10 Time Slices, and Hybrid Inference Scheme, using 1000 Samples, for ISCAS'89 Benchmark Sequential Circuits.

	3 Time slices			10 Time slices		
Circuits	E_μ	$E_{\max}$	Time(s)	E_μ	$E_{\max}$	Time(s)
s27	0.015	0.068	0.047	0.018	0.078	0.172
s208	0.014	0.234	0.719	0.006	0.197	7.532
s298	0.015	0.169	1.422	0.010	0.170	13.87
s382	0.002	0.096	2.094	0.000	0.079	21.28
s444	0.003	0.083	2.734	0.005	0.062	26.30
s526	0.003	0.044	3.922	0.001	0.081	42.28

maximum error of 0.077 in 12.39 seconds on an average for ten benchmarks. These experiments show that 1000 samples are sufficient to achieve the best accuracy-time trade-off.

In Table 5.5., we show the effect of considering increased number of time slices in the TC-LiDAG model; we consider 10 slices as opposed to 3. We observe that ten time slices do not enhance the quality of estimates. This shows that third order temporal models are good enough for our benchmarks. Interestingly, this observation matches with the observations made in [14, 16].

Table. 5.6. gives the simulation results of the sequential circuits inferenced using Probabilistic Logic Sampling. These results were obtained for 3 time-slice models of the sequential circuits with 1000 samples. The mean error and the maximum error for the circuits are almost identical to those of

Table 5.6. Experimental Results on Switching Activity Estimation by Dynamic Bayesian network Modeling for ISCAS'89 Benchmark Sequential Circuits(3 Time Slices and 1000 Samples) using Logic Sampling[48].

Circuits	E_μ	E_{max}	Time (s)	% of nodes $> \mu+2\sigma$
s27	0.028	0.092	0.016	5.88
s208	0.014	0.224	0.281	9.02
s382	0.000	0.082	0.750	7.14
s444	0.005	0.067	0.843	3.90
s526	0.002	0.048	1.234	1.84
s713	0.009	0.067	1.968	4.70
s820	0.002	0.042	2.125	4.17
s953	0.012	0.185	1.922	7.95
s1196	0.001	0.043	2.735	5.17
s1238	0.003	0.035	2.703	5.00
s1423	0.012	0.114	3.266	6.02
s5378	0.001	0.389	23.128	4.98
s15850	0.003	0.434	146.992	3.07

EPIS method. The main advantage of this method is clearly depicted through the estimation time. PLS method is, by an order of magnitude, faster than EPIS method.

CHAPTER 6

CONCLUSION

We present a unified graph based probabilistic framework for switching activity estimation method that takes into account high order spatio-temporal dependencies that are present in sequential circuits and combinational circuits. To our knowledge, this is the first work that presents a completely probabilistic approach to switching estimation in sequential circuits.

We proved that a time coupled, logically induced, directed graph structure (TC-LiDAG) can model all the independencies and is a Bayesian network, describing a joint probability distribution over all the circuit line switchings, both state and circuit lines. This model is compact, minimal and dependency preserving.

The TC-LiDAG structure also affords efficient probabilistic inference schemes. We studied both an exact inference scheme, based on the tree of cliques, and a scalable hybrid inference scheme, based on sampling and local message passing. We demonstrated the high quality of estimates formed using this model on ISCAS'89 benchmark circuits. Both the mean and the maximum errors were found to be low. The model scales well to large circuits.

The present scope of the model is limited to the zero-delay scenario, which we plan to address in future possibly by expanding the domain of the switching random variables.

In this chapter, we also present the attempts that led us to the TC-LiDAG models.

1. Our initial attempt was to feed the output state probabilities back to the input states till it converges. The convergence was not a problem, however, the system was converging to wrong values as the temporal and spatial dependencies were lost in the feedback process.

2. To overcome this, we actually unraveled the circuit for ten time slices. Each time slice were fed by independent inputs and output state line probabilities were fed to the input state probabilities of the next time slice. The random variables of interest were switching at each signals. This

model also failed as the induced dependence of the inputs and the present states are not modeled here.

3. Next, to model the input dependencies, we connected inputs of all the time slices by an identity buffer to model the induced spatio-temporal dependencies. The errors were close to 40%.

4. We modeled the dependencies between random variables that represent switching in the input between two time slices accurately and for s27 the errors were close to zero.

5. We applied the Cascaded BN approach proposed by Bhanja *et al.* [3], and we found that the estimates degenerate due to the error feedback.

6. We used the stochastic inference scheme used by Ramani *et al.* [47] for handling larger ISCAS benchmark and the any-time estimates produced accurate results for the benchmarks.

Our future direction would be addressing delay, interconnect and capacitance estimation in this problem. We intend to use this unified framework also at RT level.

REFERENCES

[1] S. Bhanja and N. Ranganathan, "Dependency preserving probabilistic modeling of switching activity using Bayesian networks," *IEEE/ACM Design Automation Conference*, pp. 209–214, 2001.

[2] S. Bhanja and N. Ranganathan, "Switching Activity Estimation of VLSI Circuits Using Bayesian Networks," *IEEE Transactions on VLSI Systems*, vol. 11, no. 4, pp. 558–567, Aug. 2003.

[3] S. Bhanja and N. Ranganathan, "Switching activity estimation of large circuits using multiple Bayesian networks," *Proceedings of ASP-DAC and 15th International Conference on VLSI Design*, pp. 187–192, 2002.

[4] R. G. Cowell, A. P. David, S. L. Lauritzen, D. J. Spiegelhalter, "Probabilistic Networks and Expert Systems," Springer-Verlag New York, Inc., 1999.

[5] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference," Morgan Kaufmann Publishers, Inc., 1988.

[6] U. Kjaerulff, "dHugin: A Computational System for Dynamic Time-Sliced Bayesian Networks," *International Journal of Forecasting*, vol. 11, pp. 89–111, 1995.

[7] "Graphical Network Interface"

URL http://www.sis.pitt.edu/~genie/genie2.

[8] S. M. Kang, "Accurate Simulation of Power Dissipation in VLSI Circuits," *IEEE Journal of Solid-state Circuits*, vol. 21, no. 5, pp. 889–891, Oct. 1986.

[9] URL http://www.hugin.com/.

[10] R. Burch, F. N. Najm and T. Trick, "A Monte Carlo Approach for Power Estimation," *IEEE Transactions on VLSI Systems*, vol.1, no. 1, pp. 63–71, 1993.

[11] R. E. Bryant, "Symbolic Boolean Manipulation with Ordered Binary-Decision Diagrams," *ACM Computing Surveys*, vol. 24, no. 3, pp. 293–318, Sept. 1992.

[12] F. Najm, S. Goel and I. Hajj, "Power Estimation in Sequential Circuits," *32nd ACM/IEEE Design Automation Conference*, pp. 635–680, 1995.

[13] A. Ghosh, S. Devadas, K. Keutzer and J. White, "Estimation of Average Switching Activity in Combinational and Sequential Circuits," *29th ACM/IEEE Design Automation Conference*, pp. 253–259, 1992.

[14] C. Y. Tsui, J. Monteiro, M. Pedram, S. Devadas, A. M. Despain and B. Lin, "Power Estimation Methods for Sequential Logic Circuits," *IEEE Transactions on VLSI Systems*, vol. 3, no. 3, pp. 404–416, 1995.

[15] G. I. Stamoulis, "A Monte-Carlo Approach fot the Accurate and Efficient Estimation of Average Transition Probabilities in Sequential Logic Circuits," *IEEE Custom Integrated Circuits Conference*, pp. 221–224, 1996.

[16] L. P. Yuan, C. C. Teng and S. M. Kang, "Statistical Estimation of Average Power Dissipation in Sequential Circuits," *34th Design Automation Conference*, 1997.

[17] Z. Chen and K. Roy, "An Efficient Statistical Method to Estimate Average Power in Sequential Circuits Considering Input Sensitivities", *ASIC Conference and Exhibit*, pp. 189–193, 1997.

[18] T.L. Chou and K. Roy, "Statistical Estimation of Sequential Circuit Activity", *Proceedings of the 1995 IEEE/ACM international conference on Computer-aided design*, pp. 34–37, 1995.

[19] J. N. Kozhaya and F. N. Najm, "Power Estimation for Large Sequential Circuits," *IEEE Transactions on VLSI Systems*, Vol. 9-2, pp. 400–406, 2001.

[20] V. Saxena, F. N. Najm and I. N. Hajj, "Estimation of State Line Statistics in Sequential Circuits," *ACM Transactions on Design Automation of Electronic Systems*, vol. 7, no. 3, pp. 455–473, 2002.

[21] A.K. Murugavel, N. Ranganathan, "Petri Net Modeling of Gate and Interconnect Delays for Power Estimation", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, pp. 921–927, 2003.

[22] J. Cheng, "Efficient Stochastic Sampling Algorithms for Bayesian Networks," *Ph.D Dissertation, University of Pittsburgh*, 2001.

[23] C. Yuan and M. J. Druzdzel, "An Importance Sampling Algorithm Based on Evidence Prepropagation," *Proceedings of the 19th Annual Conference on Uncertainty on Artificial Intelligence*, pp. 624–631, 2003.

[24] A. V. Nefian, L. Liang, X. Pi, L. Xiaoxiang, C. Mao, K. Murphy, "A Coupled HMM for Audio-Visual Speech Recognition," *Proceedings of Acoustics, Speech and Signal Processing*, 2002.

[25] K. Murphy and S. Mian, "Modeling Gene Expression Data using Dynamic Bayesian Networks," *Technical report, Computer Science Division, University of California*, Berkeley, CA., 1999.

[26] V. Mihajlovic and M. Petkovic, "Dynamic Bayesian Networks: A State of the Art," *Technical Report, TR-CTIT-01-35*, 2001.

[27] K. P. Murphy, Y. Weiss and M. I. Jordan, "Loopy belief propagation for approximate inference: an empirical study," *In Proceedings of Uncertainty in AI*, pp. 467–475, 1999.

[28] J. Geweke, "Bayesian inference in econometric models using Monte Carlo integration," *Econometrica*, pp. 1317–1339, 1989.

[29] R. Y. Rubinstein, "Simulation and the Monte Carlo method," *John Wiley & Sons*, 1981.

[30] R. Marculescu, D. Marculescu, and M. Pedram, "Probabilistic Modeling of Dependencies During Switching Activity Analysis", *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 2, pp. 73–83, February 1998.

[31] F. N. Najm, “Transition Density: A New Measure of Activity in Digital Circuits”, *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 2, pp. 310–323, February 1993.

[32] C.-S. Ding, C.-Y. Tsui, and M. Pedram, “Gate-Level Power Estimation Using Tagged Probabilistic Simulation”, *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 11, pp. 1099–1107, November 1998.

[33] L. P. Yuan, C. C. Teng, and S. M. Kang, ”Statistical estimation of average power dissipation in CMOS VLSI circuits using non-parametric techniques,” *IEEE/ACM Intl. Symp. on Low Power Electronics and Design*, pp. 73–78, 1996.

[34] F. N. Najm, R. Burch, P. Yang, and I. N. Hajj, “ Probabilistic Simulation for Reliability Analysis of CMOS Circuits”, *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, vol. 9, no. 4, pp. 439–450, April 1990.

[35] R. Burch, F. N. Najm, P. Yang, and D. Hocevar,”Pattern Independent Current Estimation for Reliability Analysis of CMOS Circuits”, *Proceedings of the 25th Design Automation Conference*, pp. 294–299, June, 1988.

[36] P. Schneider, and U. Schlichtmann, “Decomposition of Boolean Functions for Low Power Based on a New Power Estimation Technique”, *Proc. 1994 Int’l Workshop on Low Power Design*, pp. 123–128, April 1994.

[37] G. Y. Yacoub, and W. H. Ku, “An accurate simulation technique for short-circuit power dissipation based on current component isolation,” *IEEE International Symposium on Circuits and Systems*, pp. 1157–1161, 1989.

[38] M. Buhler, M. Papesch, K. Kapp, and U. G. Baitinger,”Efficient Switching Activity Simulation Under a Real Delay Model Using a Bit-Parallel Approach”, *Proc. of the European Design Automation and Test Conference*, pp. 459–463, 1999.

[39] T. H. Krodel, “PowerPlay fast dynamic power estimation based on logic simulation,” *IEEE International Conference on Computer Design*, pp. 96–100, October 1991.

[40] A. C. Deng, Y. C. Shiau, and K. H. Loh, “Time Domain Current Waveform Simulation of CMOS Circuits”, *IEEE International Conference on Computer Aided Design*, Santa Clara, CA, pp. 208–211, Nov. 7-10, 1988.

[41] D. Rabe, G. Jochens, L. Kruse, and Nebel W. Power- Simulation of Cell based ASICs: Accuracyand Performance Trade-Offs. In *Proc. of the European Design Automation and Test Conf.*, pp. 356–361, 1998.

[42] Rabe D., Nebel W. A New Approach in Gate-Level Glitch Modeling. In *Proc. of Design Automation and Test Conf. with EURO-VHDL*, pp. 66–71, 1996.

[43] F. Rouatbi, B. Haroun, and A. J. Al-Khalili,” Power estimation tool for sub-micron CMOS VLSI circuits”, *IEEE/ACM International Conference on Computer-Aided Design*, pp. 204–209, 1992.

[44] P. Vanoostende, P. Six, J. Vandewalle, and H.J. De Man,“Estimation of typical power of synchronous CMOS circuits using a hierarchy of simulators”, *IEEE Journal of Solid-State Circuits*, vol. 28, no. 1, pp. 26–39, 1993.

[45] C. Huang, B. Zhang, A. Deng, and B. Swirski, ”The design and implementation of PowerMill,” *International Symposium on Low Power Design, Dana Point, CA*, pp. 105–110, April 1995.

[46] A. Macii, E. Macii, M. Poncino, and R. Scarsi,“Stream synthesis for efficient power simulation based on spectral transforms”, *IEEE Transaction on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 3, pp. 417–426, 2001.

[47] S. Ramani and S. Bhanja, Any-time Probabilistic Switching Model using Bayesian Networks, *Accepted for publication in International Symposium on Low Power Design*, 2004.

[48] M. Henrion, “Propagation of uncertainty by probabilistic logic sampling in Bayes’ networks,”*Uncertainty in Artificial Intelligence*, 1988.