Causal Probabilistic Input Dependency Learning for Switching Model in VLSI Circuits

Nirmal Ramalingam
Dept. of Electrical Engineering
University of South Florida
Tampa, Florida-33620.
munuswam@eng.usf.edu

Sanjukta Bhanja
Dept. of Electrical Engineering
University of South Florida
Tampa, Florida-33620.
bhanja@eng.usf.edu

ABSTRACT

Switching model captures the data-driven uncertainty in logic circuits in a comprehensive probabilistic framework. Switching is a critical factor that influences dynamic, active leakage power, coupling noises in CMOS implementations. In this work, we model the input-space by a causal graphical probabilistic model that encapsulates the dependencies in inputs in a compact, minimal fashion and also allows for instantiations of the vector-space that closely match the underlying dependencies, with the constraint that the reduced vector-space captures the dependencies in the larger dataset accurately. Results on ISCAS benchmark show that average error is limited to 1.8% while we achieve a compaction ratio of 300.

Categories and Subject Descriptors: B.8.2
General Terms: Performance
Keywords: Vector compaction, Power Estimation, Cross-talk Estimation. Bayesian Networks, Probabilistic Learning.

1. INTRODUCTION

Switching model of a VLSI circuit is a comprehensive representation of switching behavior of all the signals in the circuit. At each signal we store the state at time t and the state at time $t-1$. The dependency among all the switching variable can be captured as a joint probability distribution function. One might look into switching model as a way to establish the role of data as inputs to the circuits. In essence, the switching model captures the data-driven uncertainty in VLSI circuits in a comprehensive probabilistic framework. Needless to say that, modeling inputs become an integral part of the switching model, even though a handful of prior work in power or switching analysis, has modeled inputs efficiently. Even analysis that are vector driven (simulation and statistical simulation) have mostly assumed input priors as random inputs. In this work, we take a look into modeling inputs through a causal graphical probabilistic approach which models input space in a compact way.

Note that the switching model is extremely relevant of both static and dynamic component of power as shown in Eq. 1. In this equation, P_{dg} represents the dynamic component of power at the output of a gate g. The impact of data on dynamic component of power is encapsulated in α, the singleton switching activity. The static component of power P_{sg} is dominated by $P_{leak,i}$, leakage loss in a leakage mode i. It has to be noted that each leakage mode is determined by the steady state signals that each transistor in the gate would be in. For example, in a two input (say A and B) NAND gate, the gate would have four dominant leakage mode (i=4: $A_{@0}B_{@0}, A_{@0}B_{@1}, A_{@1}B_{@0}$ *and* $A_{@1}B_{@1}$). β is the probability of each mode i. Probabilistically α and β are singleton probability of switching and joint probability of multiple signals in a gate respectively and are dependent on the input data profile. The switching model is affected by various factors such as the topology of the circuit, the input statistics, the correlation between nodes, the gate type, and the gate delays, thus making the estimation process a complex procedure.

$$\begin{aligned} P_t &= \sum_g P_{tg} = P_{dg} + P_{sg} \\ &= 0.5\alpha f V_{dd}^2 C_{load+wire} + \sum_i P_{leak,i}\beta_i \end{aligned}$$

In this work, we focus on modeling the input space and generate a probabilistic structure among the inputs that can be used to study the behavior of internal nodes. Even though estimation of singleton switching has been discussed in great depth and many of the procedures are input driven (simulative and statistical simulative), inputs are studied in a limited set of works [9, 11, 12].

Eq. 1 demonstrates the effect of input model. Let X_i be an intermediate signal which could be singleton switching variable representing switching states ($0_{t-1}0_t, 0_{t-1}1_t, 1_{t-1}0_t$ *and* $1_{t-1}1_t$), of a signal or X_i could be composite signals (A,B) in a transistor stack and can have composite states namely $A = 0_{t-1}0_t,\ B = 0_{t-1}0_t$. In estimating X_i, as shown in Eq. 1, we have two components, namely the set $P(X_i|\{I_j\})$, $\{I_j\}$ denoting the primary inputs of the circuits, where j is the jth input and $P(\{I_j\})$ a prior probability of the input space. The first component can be analyzed by probabilistic measure considering a joint pdf of the entire signals through a probabilistic framework as in [13, 5]. The second component $P(\{I_j\})$ however, is important for both stimulus-sensitive approach and stimulus-free approach where we need to model the dependency structure of the inputs and then either use it in the joint pdf of the signals for probabilistic methods or use it to generate representative specific input samples for measurement-based estimates. The theme of this work is to obtain correct priors in the input space $P(\{I_j\})$ and then to generate samples that closely emulate the behavior of the input space or to fuse the prior on to an existing probabilistic set-up. Note that, we do not concentrate on compaction, but our claim is that the samples are so close to the actual distribution that the sample requirements are really low and a high compaction ratio is automatically achieved.

GLSVLSI'05, April 17–19, 2005, Chicago, Illinois, USA.
Copyright 2005 ACM 1-59593-057-4/05/0004 ...$5.00.

$$P(X_i) = P(X_i|\{I_j\}).P(\{I_j\}) \quad (1)$$

Note that input modeling is not a simple task and can be useful in other areas like test vector and pattern generation. This work is an effort to learn a causal structure in the input data. Causal structures are common to obtain for real-life data and have been proven successful in modeling complex data-sets like gene-matching and speech processing. A causal model can encapsulate two additional independencies (induced dependency among the cause of a common child, and weakly nontransitive dependency) over and above their undirected Markov models that makes it specially attractive in reducing the structure. Even though, BN based model for interconnected logic is already proposed [5], this work is novel in learning a probabilistic causal structure from data and to our knowledge, this is the first model where causality in the data is utilized in learning a structure of uncertainty in the input-space.

Bayesian networks (BN) are directed acyclic graph (DAG) representations, whose nodes represent random variables and the links denote direct dependencies, which are quantified by conditional probabilities of a node given the states of its parents. This DAG structure essentially models the joint probability function over the set of random variables under consideration in a compact manner. The attractive feature of this graphical representation of the joint probability function is that not only does it make conditional dependency relationships among the nodes explicit but also serves as a computational mechanism for an efficient probabilistic walk generating samples.

We use a Bayesian Network to generate vectors that closely match the underlying probability distribution function such that even only a handful of vectors converges to the correct estimates using an efficient simulator (HSPICE in our case). We first convert the graphical probabilistic structure into an intermediate tree structure (junction tree) where we could use computation between neighboring nodes. Nodes of this tree structure are a subset of original variables of interest. We then use stochastic sampling in one of the cliques of the junction tree to obtain the necessary samples based on the underlying probabilistic framework. The variables that are common between the neighboring cliques are then updated. The neighboring cliques are then sampled conditional to the common variables already updated. If we represent a Bayesian network by a sample of m deterministic scenarios s=1, 2,.....m and $L_s(x)$ is the truth of event x in scenario s, then uncertainty about x can be represented by a logic sample. Also, since we start every time at an arbitrary clique and then instantiate an arbitrary node in that clique, this sampling is extremely pattern insensitive.

The contribution of this work is two-fold. First, we arrive at a probabilistic graphical model in the inputs that is (i) edge-minimal (ii) exact in terms of dependence and (iii) easy to learn. Second, it is elegant as a model and also acts as a source of generating samples from the graphical probabilistic structure that closely resemble the dependency in the inputs and a few samples converges to the mean of the underlying distribution.

The salient features of the proposed Bayesian Network (BN) learning model for inputs are as follows.

1. *It generates an edge-minimal structure that models dependency exactly under a causal data environment.*
2. *The computations are easy and learning algorithms are O(N^2) to O(N^4) in terms of number of inputs.*
3. *The dependency model of the inputs can be fused with graphical structure of the internal circuit making the estimation stimulus-free and insensitive to measurements.*
4. *The dependency model can be probabilistically efficiently sampled such that samples closely emulate the dependencies in the inputs for statistical simulation and simulation based estimation process.*
5. *The performance that is seen on the ISCAS circuits generates a maximum error of 1.8% with a compression ratio up to 300.*

2. RELATED WORK

The compaction technique used in [2] is based on fractal concepts. The correlation in the input vectors is exploited and the algorithm divides the larger vector set into smaller fractal subsets. Then the fractal subsets that are similar to a particular subset are removed from the original vector set, and thus obtaining a compacted vector set. In [9] two different techniques are proposed to accommodate temporal compatibility. The first method discards the temporal incompatibility between the pairs of consecutive vectors. The next method addresses this problem by making sure the consecutively generated vectors are temporally compatible, by proposing a greedy mechanism. In [11] Dynamic Markov Chain Modeling is used to manage correlations among adjacent bits that belong to the same vectors and correlations between successive input patterns. The Markov model of the input sequence is first derived through one-pass traversal technique and then a compacted sequence is generated, which is used to determine power consumed by the target circuit.

3. LEARNING BAYESIAN NETWORKS

In a Bayesian network, the graph, called the Directed Acyclic Graph(DAG), is the structure of the Bayesian Network, and the conditional probability distribution is called the parameter. Both the parameter and the structure can be separately learned from the data. While learning the parameter we assume we know the structure of the DAG, but in structure learning we start with only a set of random variables with unknown relative frequency distribution. Learning structure can also be done with missing data items, and hidden variables and also in the case of continuous variables as discussed in [3]. The input to learn the Bayesian network in this experiment is a database table. Each node in the network is a representative of the fields in the database. Each record in the given database is a complete instantiation of the random variables. We assume that the database table has discrete values and the data set is complete with no missing values. Also, the volume of the data set should be large enough so that reliable conditional independence (CI) tests could be performed. The conditional independence play an important role, and by using the concept of direction dependent separation or d-separation (Pearl, 1988), all the independence relations of the Bayesian network can be computed.

Definition: For a DAG $G = (V,E), X,Y \in V$ and $X \neq Y$, and $C \subset V \setminus X,Y$, we say that X and Y are d-separated given C in G if and only if there exists no adjacency path P between X and Y, such that (i) every converging arrow on P is in C or has a descendant in C and (ii) no other nodes on path P is in C. C is called the cut-set. If X and Y are not d-separated given C, we say that X and Y are d-connected given C.

If two nodes are dependent, then the knowledge of the value of one node will give us some information of the value of the other node. This information gain can be measured by using mutual information. Therefore the knowledge of mutual information can tell us about the dependency relation between two nodes. The mutual information of two nodes X_i, X_j is expressed as:

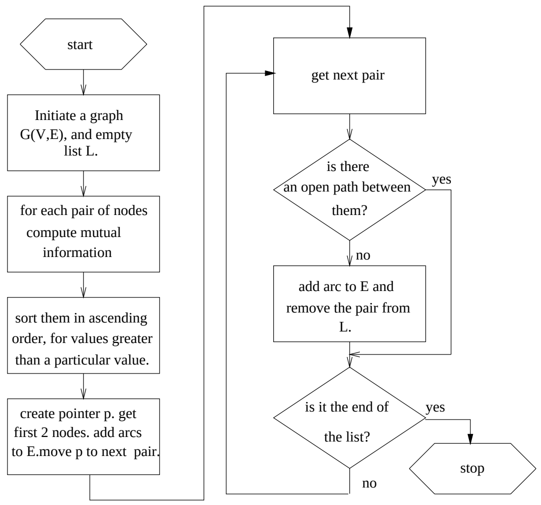
start
Initiate a graph G(V,E), and empty list L.
for each pair of nodes compute mutual information
sort them in ascending order, for values greater than a particular value.
create pointer p. get first 2 nodes. add arcs to E.move p to next pair.
get next pair
is there an open path between them?
yes
no
add arc to E and remove the pair from L.
is it the end of the list?
yes
no
stop

Figure 1: Step 1: Drafting

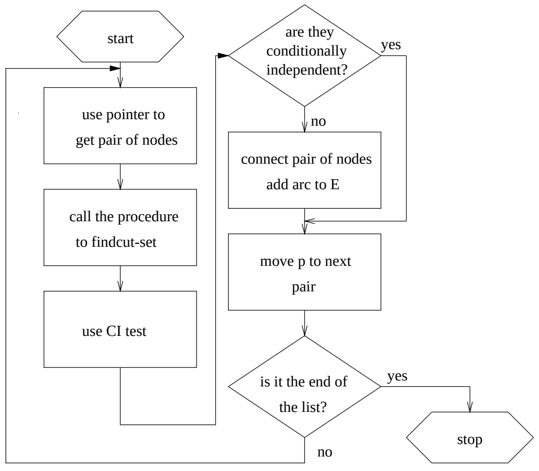
start
use pointer to get pair of nodes
call the procedure to findcut-set
use CI test
are they conditionally independent?
yes
no
connect pair of nodes add arc to E
move p to next pair
is it the end of the list?
yes
no
stop

Figure 2: Step 2: Thickening

$$I(X_i,X_j)=\sum_{x_i,x_j}P(x_i,x_j)\log\frac{P(x_i,x_j)}{P(x_i)P(x_j)} \qquad (2)$$

and the conditional mutual information is defined as

$$I(X_i,X_j\mid C)=\sum_{x_i,x_j,c}P(x_i,x_j,c)\log\frac{P(x_i,x_j\mid c)}{P(x_i\mid c)P(x_j\mid c)} \qquad (3)$$

where C is a set of nodes. When $I(X_i,X_j)$ is smaller than a certain threshold ε, we say that X_i,X_j are marginally independent. When $I(X_i,X_j\mid C)$ is smaller than ε, we say that X_i,X_j are conditionally independent given C.

4. ALGORITHM FOR LEARNING BAYESIAN NETWORK GIVEN NODE ORDERING

The algorithm constructs a Bayesian network based on dependency analysis from a database table as input. A similar work is the Chow-Liu algorithm as explained in [1].

There are three steps, that the algorithm performs, namely drafting, thickening and thinning as shown in [4].

Step 1: Drafting: A flow chart representation of the steps is shown which is self explanatory.

This step tries to find a draft which is more like the final model as much as possible by only using pairwise mutual information tests without involving conditional independence tests. The result obtained at the end of the step can be anything from an empty graph to a complete graph without affecting the final graph at the end of all the steps of the algorithm. The draft learning procedure comes to a stop when every pairwise dependency is expressed by an open path in the draft.

Step 2: Thickening: The steps here are again easily explained with the help of another flow chart.

Some arcs can be wrongly added, because some needed arcs may be missing and they hinder finding a proper cut-set.

Step 3: Thinning: For the two nodes under consideration, if there are other paths, other than the arc, then the arc is temporarily removed from E, and the procedure is called again to find a cut-set that can d-separate the two nodes. Now given the cut-set the CI test is done to find out whether the two nodes are conditionally independent. If so, the arc is permanently removed or else it is added back. This step is basically to find the arcs which may be been wrongly added in the previous steps. This step makes sure all the adds arcs are needed and there are no extra ones, and the end of the step results in the I-map of the model.

Stochastic sampling: Probabilistic sampling is performed inside the cliques of the junction tree to obtain the necessary samples. The steps involved in formation of the junction tree is the creation of a moral graph, and the process is called compilation, then the moral graph is triangulated. The clique set is identified and the junction tree of cliques is formed. Given a Bayesian network, a moral graph is obtained by 'marrying parents', that is, adding undirected edges between the parents of a common child node. Before this step, all the directions in the DAG are removed. The moral graph is said to be triangulated if it is chordal. The undirected graph G is called chordal or triangulated if every one of its cycles of length greater than or equal to 4 possesses a chord [8], that is we add additional links to the moral graph, so that cycles longer than 3 nodes are broken into cycles of three nodes. The junction tree is defined as a tree with nodes representing cliques (collection of completely connected nodes) and between two cliques in the tree T there is a unique path. We employ stochastic simulation approach to make probabilistic inferences in large multiply connected networks. If we represent a Bayesian network by a sample of m deterministic scenarios s=1, 2,.....m and $L_s(x)$ is the truth of event x in scenario s, then uncertainty about x can be represented by a logic sample.

5. EXPERIMENTAL RESULTS

This model proves to be an efficient technique for power estimation by providing a minimum number of samples that bridges the gap between simulative and probabilistic techniques. This is achieved by an efficient model of Bayesian network, by learning the structure from the input data. The learned structure of the Bayesian network represents the graphical structure in the input data. The learned structure can now be used to generate any number of vectors. The generation of the vector set is done efficiently by exploiting the sampling techniques applied on Bayesian networks. Sampling is done to derive vectors in any needed compaction ratio. The reduced vector set is then fed into HSPICE to get an accurate esti-

Table 1: Power Estimates of ISCAS '85 Benchmark Suite.

Circuit	Inputs	60K	Compaction Ratio			
			40	80	200	300
C17	5	2.9973	2.9826	2.9889	2.9829	2.9799
C432	36	4.1304	4.1669	4.1980	4.1474	4.1545
C499	41	3.0005	2.9972	3.0177	3.0927	3.1116
C1355	41	3.2257	3.2348	3.2478	3.3319	3.3245
C1908	33	5.3925	5.3634	5.4259	5.4095	5.3511
C3540	50	1.5969	1.5875	1.5866	1.5499	1.5769
C6288	32	2.2521	2.3112	2.3116	2.3142	2.3197
Avg	%	Error	0.77	1.02	1.83	1.86

mate of the average power. The power estimate is then compared with the value got from simulating the large vector set. If a is the original length of the vector and b is the compacted length, then a/b is called the compaction ratio (CR). We have achieved very high compaction ratios of up to 300 which have never been achieved before.

We have fixed the larger vector set to be 60,000 highly-correlated input patterns. To generate input patterns that are most likely, we consider the following. At RTL, the logic blocks are interconnected and hence output of one block is fed as an input to another. To simulate this effect, we sampled the outputs of one benchmark circuit and use this data as the input space of another benchmark. For example to obtain input vector space for C432 the circuit C3540 was simulated. The process is mounted on a 32-bit windows systems on PC and can be run on Windows 9x and XP versions. Since the construction engine is an ActiveX DLL, it can be integrated into other belief network, data mining or knowledge base systems. The learning time was less than a few seconds for all the benchmarks.

Table 1 shows the power estimates of the ISCAS '85 benchmark suite. The second column shows the number of inputs in each circuit and the Bayesian network model has the number of nodes equal to the number of inputs. So the largest Bayesian network used was 50 for C3540 and the smallest is of 5 nodes for C17. It is seen that as the compaction ratio increases the error% increases, but it is not that significant. Also, the error was only 0.77% for CR 40 with a maximum error of 1.86% for CR 300.

The tables 2 and 3 show the joint probability of two arbitrary nodes for two benchmark circuits for both the bigger vector set and the compacted one with the compaction ratio of 40. These values denote the joint probability of the two nodes in consideration. HSPICE is used to print the voltage values for the particular input vectors of the two nodes, which are then changed to their switching values. From these voltage values the joint probability is calculated by finding the number of occurrences. The main idea of this part of the result is to estimate the probability of worst-case crosstalk, but the table also gives an estimate of the probability of leakage occurring in the nodes.

The tables 2 a and b were plotted for the nodes 300 and 330 which are the inputs to a gate selected at random for original and reduced vector-sets and the probabilities match closely with each other. Note that here, node 300 is steadily remaining at zero which is not true if you assume random inputs. Even though , worst crosstalk probability is $P(300@01,\ 330@10) = 0$, but we have significant active leakage $P(300@00,\ 330@11) = .56$. Similar results are also shown for another set for c499 in table 3 where we have a 7% probability of worst case cross-talk.

Thus, in this work, we provide a graphical probabilistic causal model for the input space for efficient estimates of the switching

Table 2: Joint Probability Switching Estimate for Nodes 300 and 330 of C432 for(a) 60K vectors (b) for compression ratio =40

Node 300	Node 330			
State	00	01	10	11
00	0.06	0.19	0.19	0.54
01	0	0	0	0
10	0	0	0	0
11	0	0	0	0

(a)

Node 300	Node 330			
State	00	01	10	11
00	0.05	0.19	0.19	0.56
01	0	0	0	0
10	0	0	0	0
11	0	0	0	0

(b)

Table 3: Joint Probability Switching Estimate for Nodes 557 and 558 of C499 (a) 60K vectors (b) for compression ratio =40

Node 557	Node 558			
State	00	01	10	11
00	0.05	0.06	0.06	0.06
01	0.06	0.06	0.07	0.07
10	0.06	0.07	0.06	0.07
11	0.07	0.06	0.07	0.07

(a)

Node 557	Node 558			
State	00	01	10	11
00	0.06	0.06	0.06	0.06
01	0.06	0.06	0.07	0.06
10	0.06	0.07	0.06	0.07
11	0.07	0.06	0.07	0.07

(b)

models that not only models singleton switching but also models joint switchings of neighboring nodes.

6. REFERENCES

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

[2] R. Radjassamy, J. D. Carothers "Vector Compaction for Efficient Simulation-Based Power Estimation", *IEEE Symposium on IC/Package Design Integration*, pp.138-142, Feb 1998.

[3] R. E. Neapolitan, "Learning Bayesian Networks", *Prentice Hall Series in Artificial Intelligence*, 2004.

[4] J. Cheng, D. Belland W. Liu, " Learning Bayesian Networks from Data: An Efficient Approach Based on Information Theory", *Technical Report, Department of Computer Science, University of Alberta*, 1998.

[5] S. Bhanja, N. Ranganathan, "Switching Activity Estimation of VLSI Circuits using Bayesian Networks", *IEEE Transactions on VLSI Systems*, pp. 558- 567, Feb. 2003.

[6] J. Cheng, D. Bell. W. Liu, "Learning Belief Networks from Data: An Information Theory Based Approach", *Proceedings of the Sixth ACM International Conference on Information and Knowledge Management*, 1997.

[7] P. Spirtes, C. Glymour, R. Scheines, "Causation, Prediction and Search", *MIT Press, 2nd Edition*, 2000.

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

[9] C. Tsui, R. Marculescu, D. Marculescu, M. Pedram, "Improving the efficiency of power simulators by input vector compaction",*IEEE/ACM DAC*, pp. 165-168, 1996.

[10] R. Marculescu, D. Marculescu, M. Pedram, "Sequence Compaction for Power Estimation: Theory and Practice", *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol.18, No.7, pp. 973-993, July 1999.

[11] R. Marculescu, D. Marculescu, M. Pedram, "Vector Compaction Using Dynamic Markov Models", *IEICE Trans. on Fundamentals*, vol. E80A, No.10 pp. 1924-1933, 1997.

[12] A. Macii, E. Macii, M. Poncino, R. Scarsi "Stream Synthesis for Efficient Power Simulation Based on Spectral Transforms ", *Proceedings of the International Symposium of Low Power Electronics and Design*, pp.30-35, 1998.

[13] 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. 17-2, pp. 73-83, February 1998.

[14] URL `http://www.hugin.com/`