Node Sampling Technique to Speed Up Probability-Based Power Estimation Methods

Hoon Choi, Hansoo Kim, In-Cheol Park, Seung Ho Hwang and Chong-Min Kyung
Department of Electrical Engineering
Korea Advanced Institute of Science and Technology, Taejon, Korea

Abstract

We propose a new technique called *node sampling* to speed up the probability-based power estimation methods. It samples and processes only a small portion of total nodes to estimate the power consumption of a circuit. It is different from the previous speed-up techniques for probability-based methods in that the previous techniques reduce the *processing time for each node* while our method reduces the *number of nodes actually processed*. In addition, it is also different from the previous statistical sampling simulation techniques for simulation-based methods in that the previous methods sample the *input vectors* while our method samples the *nodes in the network*. The experimental results are very encouraging. The proposed method shows on the average more than 80% and 60% reductions of simulation run time under 20% and 5% error bounds, respectively.

1. Introduction

The continuing decrease in feature size and the corresponding increase in the number of devices on a chip, combined with the growing demand for portable communication and computing equipments, have made the power consumption one of the major concerns in VLSI circuits and systems design. To minimize the power consumption, we need accurate and efficient power estimation tools.

Existing power estimation techniques at the gate level can be classified into two groups [1]: *probability-based* and *simulation-based methods*. In this paper, we deal with the probability-based techniques. The probability-based techniques rely on the probability information (such as the mean activity of the input signals and their correlations) about the input stream to estimate the internal switching activities of the circuit. Though these are more efficient than the simulation-based methods, they are less accurate. There have been several works [3-8] to improve the accuracy by taking into account the effects coming from simultaneous input changes, spatial and temporal correlations, glitch generation and propagation under various delay models, etc. However, the accuracy improvement is obtained at the expense of estimation speed degradation. Since the estimation speed is one of the most important merits of the probability-based method, we have to devise a technique to keep it efficient even with the treatments for accuracy.

We have two choices for the reduction of simulation run time of the probability-based method. One is to reduce the *processing time for each node*, and the other is to reduce the *number of nodes that are actually processed*. For the former, there have been several works with the approach of using a local BDD in place of a global BDD that significantly reduces the run time for each node [9-11]. However, for the best of our knowledge, there has been no reported work for the latter. In this paper, we investigate the second approach; we focus on how to reduce the number of actually processed nodes in order to speed up while losing the accuracy minimally. Briefly speaking, we employ a statistical sampling technique to sample and process only a small portion of the total nodes, and then elicit the total power consumption from the sample's power consumption. We call this *node sampling* technique. Notice that the proposed technique can be applied together with the previous methods of reducing the processing time for each node in order to reduce the simulation run time further because they are independent of each other.

At this point, we would like to compare the proposed method with the statistical sampling techniques used in [12], [13], which are for the simulation-based power estimation methods. The simulation-based power estimation methods use conventional logic simulators with input vectors (randomly generated or user-given) at the circuit inputs to monitor the power dissipation. Sampling techniques are used to sample input vectors from the total input vector sequences as shown in Fig. 1-(a). In contrast, the proposed method samples nodes from the network, not the input vectors, to speed up the probability-based power estimation methods. This is illustrated in Fig. 1-(b). In short, the proposed technique and the previous ones are different in 1) for which estimation method the statistical sampling technique is applied, i.e., simulation-based versus probability-based, and 2) what are sampled, i.e., input vectors versus nodes in the network.

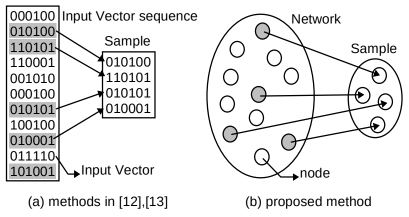

Fig. 1: Comparison with previous methods

The target probability-based method to which we apply the *node sampling* technique in this paper is the *symbolic simulation* method [2] (Notice that this is just for a more specific explanation of the proposed technique. The node sampling technique can be similarly applied to other probability-based power estimation methods). It builds a symbolic network composed of symbolic nodes each of which represents a possible glitch of a gate in a logic circuit. Though, the estimates provided by the symbolic simulation are accurate, it has not been widely used due to the slow estimation speed. In this work we show that the *node sampling* technique can reduce the CPU-time of the symbolic simulation method significantly.

The rest of the paper is organized as follows: Section 2 briefly reviews the preliminaries: statistical sampling simulation and symbolic simulation method. In Section 3, the details of the proposed method is explained. We present the experimental results in Section 4 and Section 5 concludes the paper.

2. Preliminaries

2.1. Statistical Sampling Simulation

In a statistical sampling simulation, we build samples from a population and then simulate them to obtain the property of samples from which the property of the population is elicited [14]. According to the *central limit theorem*, as the sample size approaches to infinity, the density of samples' property tends to be a normal curve. In practice, a sample size of 30-50 ensures normal sample density for well-behaved combinational circuits. For a desired error tolerance in the estimate ε, with a given confidence level $1 - \alpha$, a sample mean η_T, and a sample standard deviation s_T, the number of required samples N can be estimated as $N > \left(\frac{t_{\alpha/2} s_T}{\varepsilon \eta_T} \right)^2$ where $t_{\alpha/2}$ is obtained from the t distribution with $(N - 1)$ degrees of freedom.

In selecting samples, the stratified random sampling is preferred to the simple random sampling because of the following advantages [14]: 1) The sample variance decreases, and thus the simulation run time can be decreased, and 2) the sample distribution is more likely to be a normal distribution. The stratified random sampling partitions the population into disjoint subpopulations so that the property within each subpopulation is more homogeneous than in the original population. Then, a sample is made by selecting some elements from each subpopulation.

2.2. Symbolic Simulation Method

The concept and algorithm of the symbolic simulation method was proposed in [2]. It constructs the Boolean functions describing the gate outputs at discrete time points implied by the delay model under a pair of input vectors. The inputs to the created Boolean functions are the circuit input lines at time instances 0^- and ∞. For each gate output i, the Boolean function $f_i(t+\tau)$ evaluates to 1 if the gate output is 1 at time $t+\tau$. Boolean functions describing the logic values of a node at two consecutive time instances are XORed to determine whether a transition occurs at a boundary between time instances. The output of the XOR gate evaluates to one exactly when the node makes a transition between the two time instances. Summing up the signal probabilities of these XOR gates gives the average switching activity. We call this Boolean network *symbolic network*. The node in the symbolic network, i.e., Boolean function $f_i(t+\tau)$, is called *symbolic node*. In calculating the signal probability of a symbolic node, BDD is used. The minimum number of symbolic nodes used as variables of a BDD are called *support-nodes* [9-11].

3. Proposed Method

3.1. Overview

In this section, we explain the overall procedure of applying the node sampling technique to the symbolic simulation method with the help of Fig. 2 which shows one pass of the proposed method.

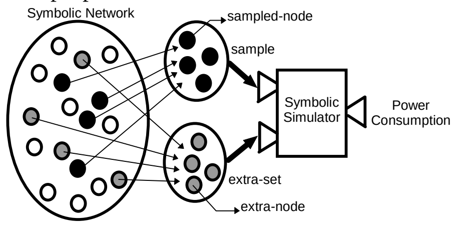

Fig. 2: One pass of proposed method

First, the given logic circuit is translated into a symbolic network considering the given delay model. Second, a sample composed of n symbolic nodes is built using a sampling technique which will be explained later. The nodes in a sample, black circles in Fig. 2, will henceforth be called *sampled-nodes*. Third, an extra-set is built, which consists of the minimum number of nodes that should be processed by the symbolic simulator to estimate the power consumption of the sampled-nodes. Those nodes in the extra-set, shaded circles in Fig. 2, will be called *extra-nodes*. The necessity and details of this step will be explained in Section 3.2. Fourth, the symbolic simulator processes the sampled-nodes and extra-nodes, and computes the power consumption of the sample. Lastly, the stopping criterion is tested with the mean and the variance of the power samples. If it is unsatisfied, we repeat the above procedures from the second step. If satisfied, we convert the mean of the power samples into the total power of the entire network and finish the simulation. The pseudo code of the overall procedure is shown in Fig. 3.

```
/* $N_{LC}$ : given logic circuit */
/* $N_{SYM}$ : symbolic network */
$N_{SYM}$ = symbolic network translator($N_{LC}$);
iter = 0; /* # of iteration (# of samples) */
repeat {
    {$n_{s1}$, $n_{s2}$, …, $n_{sn}$} = sampling($N_{SYM}$); /* $n$ sampled nodes */
    {$n_{e1}$, $n_{e2}$, …, $n_{er}$} = extra node search($N_{SYM}$, {$n_{s1}$, $n_{s2}$, …, $n_{sn}$});
    symbolic simulation($N_{SYM}$, {$n_{s1}$, $n_{s2}$, …, $n_{sn}$}, {$n_{e1}$, $n_{e2}$, …, $n_{er}$});
    $P_S$ = power calculation of a sample($N_{SYM}$, {$n_{s1}$, $n_{s2}$, …, $n_{sn}$});
    calculate $\eta_P$, $s_P$; /* mean and STD of power samples */
    Flag = TRUE if stopping criterion is met;
    iter = iter + 1;
} until (Flag == TRUE && iter > LowLImit);
$P_{TOTAL}$ = convert($\eta_P$); /* Total power */
```

Fig. 3: Pseudo code of overall procedure

3.2. Algorithm details

We use a stratified random sampling to select symbolic nodes for a sample. Our stratification scheme is based on the results of [13]. It suggested that the following scheme is efficient: 1) The number of strata, k, is set to 10. An increase in k beyond 10 is seldom profitable. 2) The construction of strata on the basis of equalization of W_iS_i and equal sample size allocation to the strata leads to a good stratification. (W_i is the ratio of the size of the i'th stratum to the size of total population, and S_i is the standard deviation of the i'th stratum). We complied with the first one, i.e., k = 10. However, to follow the second one, we have to use low-cost prediction. Though we can employ fast estimation techniques like *TSTD* in [8] or zero-delay estimation as a predictor, it may increase the total run time. Hence, in this work we do not use such a scheme but employ other simple but efficient scheme. We first group all the symbolic nodes corresponding to a node in the logic circuit into one group as shown in Fig. 4. In the figure, the first node of a logic circuit has four possible transitions. Hence it is mapped into a group of four symbolic nodes which constitute the first group, G_1. Stratification is done in such a way that the symbolic nodes in a group belong to the same stratum. The rationale of this grouping is as follows. One of the largest factors to the variance of individual node's power consumption in a logic circuit is the glitch power. All the glitches of a node in a logic circuit are transformed and represented by a group of symbolic nodes where each symbolic node represents the possible glitch at each time instance. Hence, if we make a sample by selecting evenly from different groups of symbolic nodes corresponding to a different node of the logic circuit, the sample tends to include a portion of the glitch power evenly from all the nodes of the logic circuit. This is beneficial in reducing the sample variance.

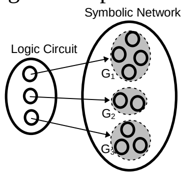

Fig. 4: Grouping of symbolic nodes

Then, we arrange the symbolic nodes into a *nodeList* where the symbolic nodes in the same group are adjacent to each other and a group corresponding to the primary input (PI) side node of a logic circuit precedes that corresponding to the primary output (PO) side node. Then the stratification is performed in such a way that the number of groups contained in each stratum is adjusted to include as a similar number of symbolic nodes as possible, and the number of strata is equal to 10 as mentioned earlier. This process is illustrated in Fig. 5 where G_i is the i'th group and S_i is the i'th stratum. Because of the listing order of the nodeList, i.e., from PI side to PO side, a stratum includes the groups corresponding to the nodes of the logic circuit whose depths from PI side are similar to each other. This is important due to the following reason. In general, PO side node in a logic circuit has more glitches than PI side one. Thus, if we sample the symbolic nodes corresponding to the nodes of the logic circuit mainly from one side, the sample can not represent the property of the entire symbolic network properly. The ordering heuristics prevents such a sampling, and enables us to build a sample that are selected evenly from both sides because the ordering makes the similar depth groups be included in the same stratum. Note that more sophisticated stratification techniques can be applied for better sampling for other target probability-based methods. For example, as mentioned before we can use a low-cost predictor and the sophisticated stratification method of [13] for better sampling. However, in general, appropriateness depends on the target probability-based method and the detailed selection guide is beyond the scope of this paper.

We build a sample composed of n symbolic nodes by selecting equal number of symbolic nodes randomly from each stratum. For such a sampling to be a valid one, each power sample has to be an independent random variable. To meet this requirement, we restart the random number generation for each sampling. Then, the sampled-nodes become independent of the previous ones, and hence the sample also does. Thus, the corresponding power sample can be independent of the previous ones.

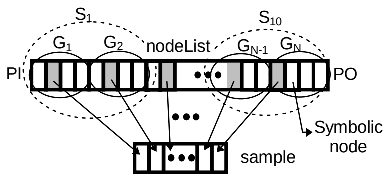

Fig. 5: Stratification

After selecting n symbolic nodes for a sample, we search the nodeList starting from PO side to PI side in order to find extra-nodes. The extra-nodes are the minimum number of nodes whose signal probability has to be calculated to estimate the power consumption of a sample: As explained in Section 2, we have to know at least the signal probability of support-nodes to estimate the power consumption of a node. The procedure is as follows: During the search from PO side to PI side, if we meet a sampled-node or an extra-node that was not processed in the preceding iterations, we examine all the support-nodes of the node. If the support-node is neither a sampled-node nor an extra-node and was not processed in the preceding iterations, it becomes an extra-node. This step repeats until we search all the nodes in the nodeList. The extra-nodes are used as the variables of a BDD for calculating the power consumption of a sampled-node.

After selecting sampled-nodes and finding extra-nodes, we use the symbolic simulator to estimate the power consumption of a sample. In the symbolic simulation, only the sampled-nodes and extra-nodes are processed. More precisely, among them only those that have not been processed in the preceding iterations are actually processed. For the nodes that have been processed in the preceding iterations, the results in the preceding iterations are used. Note that as the iteration goes on, the number of nodes whose power consumption has already been calculated in the preceding loops increases. Therefore, the number of nodes actually processed in each iteration usually decreases as iteration goes on.

From the symbolic simulation results, we obtain the power consumption of a sample, i.e., a power sample. We calculate the mean and standard deviation of all the power samples, and then we test the stopping criterion. Note that the stopping criterion is tested only after a predefined minimum number of iterations, in order to prevent a wrong premature stopping. If the stopping criterion is not met, one more iteration is started. If met, we convert the mean of the power samples, η_S, to the total power consumption of the network

4. Experimental Results

We implemented the proposed *node sampling* technique in the symbolic simulation package of the SIS program and used the ISCAS'85 benchmark circuits as our input circuits. In all the symbolic simulations, the local BDD

[11] is used in place of the global BDD. Sample size is set to 30 based on the result in [13].

First, we show the simulation run time ratio and estimation error obtained by the proposed method in Fig. 6. We experimented with two sets of user-given constraints: 1) 20% error bound with 95% confidence level, and 2) 5% error bound with 99% confidence level.

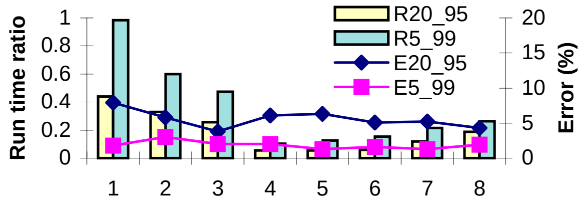

Fig. 6: Estimation error and simulation run time ratio of the proposed method (R20_95, E20_95: run time ratio and estimation error respectively under 20% error bound with 95% confidence level; R5_99, E5_99: run time ratio and estimation error respectively under 5% error bound with 99% confidence level).

Under 20% error bound with 95% confidence level, the average estimation error was 5.6%, which satisfies the given error bound well, and the average simulation run time ratio is 0.188 which means that the proposed method uses only about 20% of the simulation run time of the conventional symbolic simulation, i.e., 80% reduction. For 5% error bound with 99% confidence level, the average estimation error was 1.8% and the average simulation run time ratio is 0.365, i.e., more than 60% reduction. The average error is much less than the given error bound. This can be explained from the normal deviation, *z*, of a normal curve under the given confidence level [14]. The normal deviation under 99% confidence level is 2.57. Thus, the theoretic average error for 5% error bound is 1.94% which is almost the same with the experimental result, 1.8%.

Second, we show the typical convergent behavior of the proposed method in Fig. 7. The figure shows that the power estimate from three different runs converging to the average power as iteration goes on.

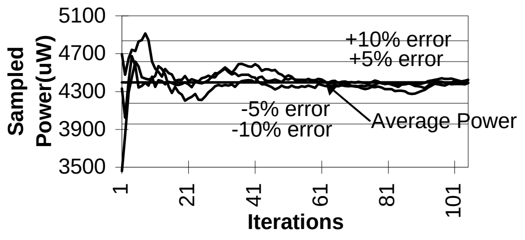

Fig. 7: Convergent behavior for C880

Lastly, we show the effect of the stratified random sampling compared to the simple random sampling in Table 1. It shows the average and the maximum reductions in percentage in the number of required samples and in the simulation run time. We can see that the stratified random sampling shows better results than the simple random sampling.

5. Conclusions

In this paper, we proposed a *node sampling* technique to speed up the probability-based power estimation methods. It samples only a small portion of the total nodes in estimating the power consumption of a circuit using the statistical sampling technique. It is different from the previous speed-up techniques for the probability-based methods and the previous statistical sampling simulation techniques for the simulation-based methods. The experimental results show that the proposed method applied to the symbolic-simulation based power estimation method significantly reduces the simulation run time under a given estimation accuracy and confidence level.

20% err, 95% conf.				5% err, 99% conf.			
# samples		sim. time		# samples		sim. time	
Avg	Max	Avg	Max	Avg	Max	Avg	Max
14	38	3	15	25	51	12	15

Table 1: Reductions in percentage obtained by the stratified random sampling compared to the simple random sampling

References

[1] F. N. Najm, "Power Estimation Techniques for Integrated Circuits," *ICCAD*, pp. 492-499, 1995.

[2] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of Average Switching Activity in Combinational and Sequential Circuits," *DAC*, pp. 253-259, 1992.

[3] L. Chou, K. Roy, and S. Prasad, "Estimation of circuit activity considering signal correlations and simultaneous switching," *ICCAD*, 1994, pp. 300-303.

[4] F. N. Najm, "Low-pass filter for computing the Transition Density in digital circuits," *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 1123-1131, Sep. 1994.

[5] Kapoor, "Improving the accuracy of circuit activity measurement," *DAC*, 1994, pp. 734-739.

[6] Mehta, M. Borah, R. M. Owens, and M. J. Irwin, "Accurate estimation of combinational circuit activity," *DAC*, 1995, pp. 618-622.

[7] R. Marculescu, D. Marculescu and M. Pedram, "Switching Activity Analysis Considering Spatiotemporal Correlations," *Proc. ICCAD*, pp. 294-299, 1994.

[8] H. Choi and S. H. Hwang, "Time-Stamped Transition Density for the Estimation of Delay Dependent Switching Activities," *ICCD*, pp. 68-73, 1997.

[9] D. I. Cheng, M. Marek-Sadowska, and K. T. Cheng, "Speeding Up Power Estimation by Topological Analysis," *CICC*, pp. 623-626, 1995.

[10] P. H. Schneider, U. Schlichtmann, and B. Wurth, "Fast Power Estimation of Large Circuits," *IEEE Design & Test of Computers*, pp. 70-78, Spring 1996.

[11] H. Choi and S. H. Hwang, "Improving the Accuracy of Support-Set Finding Method for Power Estimation of Combinational Circuits," *European Design & Test Conference (EDAC/ETC/ASIC)*, pp. 526-530, 1997.

[12] R. Burch, F. Najm, P. Yang and T. N. Trick, "A Monte Carlo Approach for Power Estimation," *IEEE Trans. on VLSI Systems*, vol. 1, no. 1, pp. 63-71, March 1993.

[13] C. S. Ding, C. T. Hsieh, Q. Wu and M. Pedram, "Stratified Random Sampling for Power Estimation," *ICCAD*, pp. 576-582, 1996.

[14] Taro Yamane, *Elementary Sampling Theory*, Prentice-Hall, 1967.