Circuit Power Estimation Using Pattern Recognition Techniques

Lipeng Cao

Somerset Design Center, Motorola Inc
Austin, TX 78729

Abstract

This paper proposes a circuit power estimation method using Bayesian inference and neural networks. Based on statistical distribution of circuit leakage power and switching energy, the entire state and transition space of a circuit are classified using neural networks into a limited few classes that represent different power consumption average values. This technique enables efficient table-lookup of circuit power of the entire state and transition space. Theoretical basis of Bayesian inference, feature extraction for neural networks of circuit leakage power and switching energy are discussed. Experiments on a wide range of circuit topologies demonstrated the robustness of the proposed method for estimating circuit leakage power of all possible states and switching energy of all possible transitions.

I. Introduction

Power consumption is becoming one of the most important factors in the design of VLSI systems in recent years due to increased integration level and higher clock frequency. Integrated circuits with high power consumption levels have stringent requirements on heat removal and management of di/dt noise. They also shorten battery life of portable electronics. Detailed and accurate power analysis on a cycle-by-cycle basis is therefore imperative not only to quantify the requirements of heat removal and di/dt noise management, but also to provide a blueprint for opportunities of reducing power consumption and mitigating di/dt noise in a design. Power consumption can be estimated at high-level [1,2], gate-level [3], and transistor-level [4], with a trade-off between estimation accuracy and simulation speed. Power estimation on a cycle-by-cycle basis is only feasible by using the gate-level or transistor-level approach. The transistor-level method provides better accuracy, but its requirement of a relatively long simulation time prevents it from being used to study a large number of test vector sequences in a large and complex design.

In this paper, we address library circuit power modeling techniques of the gate-level cycle-by-cycle power estimation methodology, in which the switching activities of the primary inputs/outputs (PI/PO) of gates in a circuit library used by a design are obtained by logic simulation. By extracting the capacitive loads at PI/PO of gates from placement/routing information of the design, cycle-by-cycle power consumption resulting from the charging and discharging of capacitors of interconnects and gates' inputs can be easily evaluated. For microprocessor and SoC designs that use a substantial number of complex and custom gates, our data indicates this accounts for no more than 40~60% of the total power consumption excluding on-chip caches. The remaining part of the power consumed internal to gates are estimated by pre-characterized circuit power models, which evaluate the power consumption of gates based on their PI/PO (state/transition) information from logic simulation at every simulation time step. An example of the commercial tools implementing such a power estimation methodology is Synopsys Prime Power [5]. Power estimation accuracy of this gate-level method depends on how well the power consumption (both leakage power and switching energy) of gates in the circuit library used by a design is modeled. Since circuit (gate) leakage power is state dependent and circuit internal switching energy is transition dependent, accurate power estimation of circuit power needs to capture the dependency on state and transition of circuit primary inputs. In general, a limited number of data points of the power consumption of a circuit are obtained by SPICE simulation using a stimulus that is generated either randomly [6,7] or by algorithmic methods. The challenge is to estimate the leakage power of *every possible state*, and the switching energy of *every possible transition*, using the limited number of SPICE data points available with the exponential growth of the state and transition space of a circuit with respect to its number of primary inputs.

Modeling of circuit power dependency on states/transitions in the entire state/transition space is very limited in [5]. This is because of its use of Boolean equations to group states/transitions and its limitation of only modeling single-pin switching, which is a very small fraction of the possible transitions for a majority of the circuits used in microprocessor and SoC designs. The use of polynomial equations is proposed in [8], but it is unclear how effective the proposed method is on modeling state/transition dependent circuit power consumption for a wide variety of circuit topologies. In this paper, we propose to use Bayesian inference and pattern recognition techniques using neural networks to solve this problem. The idea is to use statistical information of the available SPICE power data points of a circuit to characterize the correlation between state/transition patterns and power consumption values of the circuit. Such correlated pattern information is further used to predict the power consumption of *any single* seen and unforeseen state/transition in the *entire* state/transition space of the circuit. It should be noted that the issue addressed in this paper is different from other probabilistic switching activity estimation approaches (e.g. [9,10]), in which switching probabilities at PI/PO of gates of a design for a *sequence* of events in time domain is estimated.

0-7803-7607-2/02/$17.00

II. Problem Specification

The leakage power and internal switching energy of a circuit observe certain statistical distribution properties that are unique to the circuit. The values of leakage power and switching energy can vary by orders of magnitude from one state/transition to another. At the same time, many states have similar leakage power, and many transitions have similar switching energy. A limited few average values of a circuit's leakage power and switching energy can be derived from clustering its spectrum of leakage power and switching energy collected from SPICE simulation of a randomly generated test vector sequence [6,7] for efficient table-lookup of the circuit's power consumption. We would like to partition (classify) the *entire* state and transition space of the circuit with respect to these few limited average values. In other words, we need a mechanism to map each one of the possible states to one of the leakage power average values, and map each one of the possible transitions to one of the average switching energy values in such a way that the power estimation error is minimized.

III. Circuit Power Estimation Using Bayesian Inference

We solve the partitioning problem specified in section II through Bayesian inference. A more rigorous theoretical treatment of Bayesian inference can be found in [11,12]. In this section, we illustrate the key concepts of Bayesian inference and its application to circuit power estimation using the example of estimating the internal switching energy of the 8-to-1 mux circuit (mux8) shown in Fig. 1 (a). Procedure for estimating circuit leakage power is very similar.

Bayesian inference is based on Bayes' theorem:

$$P(C_k \mid x) = \frac{P(x \mid C_k) \bullet P(C_k)}{P(x)} \qquad (1)$$

Here, C_k denotes a class k, which represents a specific average power value. $\boldsymbol{x}$ is a feature vector that characterizes the states and transitions of a circuit. $P(\boldsymbol{x})$ is the prior probability. This is the probability that $\boldsymbol{x}$ occurs, and it functions as a normalization factor. $P(C_k)$ is the prior probability that the average power value identified by C_k is used. $P(\boldsymbol{x}|C_k)$ is the conditional probability. This is the probability that $\boldsymbol{x}$ occurs, given that C_k occurs. $P(C_k|\boldsymbol{x})$ is the posterior probability. This is the probability that C_k occurs, given that $\boldsymbol{x}$ occurs.

Power estimation using Bayesian inference involves a number of steps, which are illustrated in Fig. 1:

- Collect statistical distribution of circuit power from randomly generated test vectors. Fig. 1 (b).
- Cluster the statistical distribution into a limited few classes (average values). Fig. 1 (c).
- Extract feature vector $\boldsymbol{x}$ for circuit switching power.
- Evaluate $P(C_k)$, $P(\boldsymbol{x}|C_k)$ using the clustered statistical distribution information. Fig. 1 (c), (d), (e).
- For a transition t in the transition space, use Bayes' theorem to calculate $P(C_k|\boldsymbol{x})$. Fig. 1 (f).
- Assign an average switching energy value to the transition t based on calculated $P(C_k|\boldsymbol{x})$.

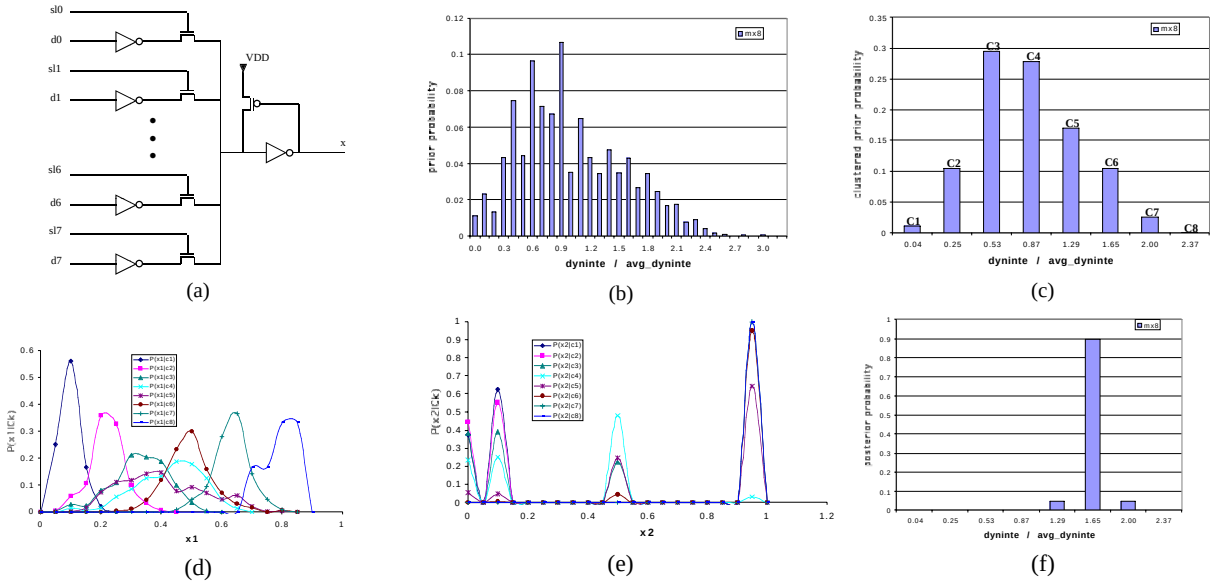

Fig 1. Illustration of power estimation using Bayesian inference. dyninte and avg_dyninte denote internal switching energy and average switching energy, respectively. (a) schematic diagram of 8-to-1 mux (mux8). (b) statistical distribution of internal switching energy. (c) clustered statistical distribution of internal switching energy (8 classes or clusters). (d) conditional probability distributions of $P(x_1|C_k)$. (e) conditional probability distributions of $P(x_2|C_k)$. (f) using Bayes' theorem to calculate the posterior probabilities $P(C_k|\boldsymbol{x})$ of a specific transition t.

Feature vector $\boldsymbol{x}$ is extracted by examining the circuit topology and identifying major sources of internal switching energy. We need to encode the transition of the primary inputs into key features that represent the major sources of the internal switching energy of the circuit. From the schematic diagram in Fig. 1 (a), it is not difficult to identify that there are two key components of the switching energy: the bank of input inverters and the output inverter. The common element is the circuit primitive inverter. We can encode the switching activity of the inverter as: trans(0) = 0.0, trans(1) = 0.1, trans(r) = 0.5, and trans(f) = 1.0. Here, trans(x) is the encoding function. And 0, 1, r, f denotes the four possible transitions (including stationary transitions). The encoded values represents the relative amount of switching energy associated with these 4 possible transitions. We extract two features:

- x_1: input data transition encoding, with encoded value as Σ(inverter encoding of each input inverter) / 8.0.
- x_2: inverter encoding of the output inverter, with the input transition of the output inverter derived from function simulation of the primary input transitions.

In Fig 1 (f), we interpret the data as: the transition *t* is most likely to be mapped into C_6, with a small probability to be mapped into C_5, C_7, and it is very unlikely to be mapped into C_1, C_2, C_3, C_4, C_8. Therefore, we can assign the average switching energy value represented by C_6 as the switching energy of the circuit for the transition *t*.

Bayes' theorem therefore allows the use of statistical information from a set of sample data, Fig. 1 (a)-(e), to evaluate the likelihood of internal switching energy of *any* possible transitions, Fig. 1 (f).

IV. Neural Networks for Power Estimation

It has been shown [11,12] that neural networks have the underlying mathematical property of Bayesian inference. We directly adopt the techniques of solving the 1-of-c classification problem in the area of neural networks to address the circuit power estimation problem.

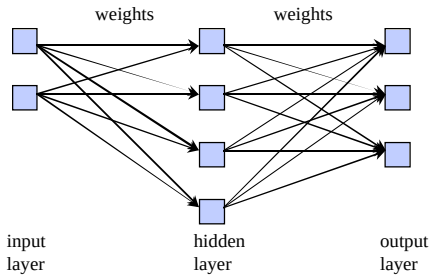

Fig: 2. Feed-forward neural network

We use the feed-forward neural network shown in Fig. 2. It's a direct acyclic graph. The graph node is called unit. Each unit has a value and an activation function associated with it. Each graph edge has its own weight. The value of a unit is calculated by its activation function based on the weights of incoming graph edges and the values of units these incoming graph edges are connected to. A neural network needs to be trained and validated before it can be used. The weights in the network are adjusted during network training. Training and validation data are derived from statistical sampling of circuit leakage power and switching energy via SPICE simulation. Commonly used training and validation techniques of neural networks are used in our approach.

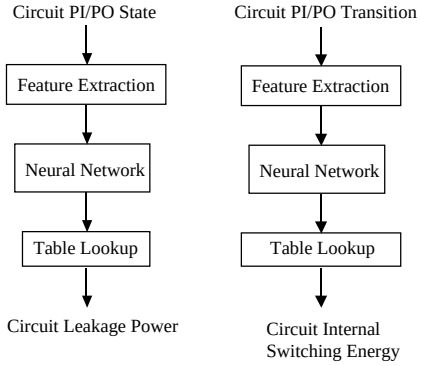

Fig: 3. Block diagram of circuit power estimation using neural networks in table-lookup

Each input unit is associated to a distinctive feature of circuit state/transition. Each output unit is associated to a predefined class of circuit leakage power/switching energy. The number of output units is equal to the number of classes created for the circuit leakage power or switching energy. Each class represents an average power consumption value. The number of hidden units is adjusted to meet the requirements of prediction accuracy and network complexity. The more hidden units there are, the more complex the network is, and the more accurate the solution of the classification problem tends to be. It is discussed in [11,12] that when logistic sigmoid and/or softmax activation function(s) are used, the values of the output units can be interpreted as posterior probabilities.

The block diagrams for our approach in estimating circuit leakage and internal switching power is shown in Fig. 3. There are several options to perform table-lookup once the posterior probabilities are calculated using neural networks. One option is the maximum likelihood approach. This selects the average value that is associated to the class with the largest posterior probability. The other option is to use weighted averages of the predefined average values. The weighting factors are the posterior probabilities of each class. In practice, a combination of these two options can be judiciously adopted.

V. Feature Extraction

The prediction accuracy of the power estimation method discussed in this paper largely depends on the quality of the feature extraction for circuit leakage and switching power. Feature extraction is performed by encoding the state of a circuit in the case of leakage power estimation, or by encoding the transition of a circuit in the case of switching energy estimation. A properly selected feature x should produce two or more distinctively identifiable conditional probability distributions $P(\boldsymbol{x}|C_k)$, as those shown in Fig. 1 (d) and (e). Neural networks use such conditional probability distributions to make decisions on assigning a state or transition to the right class, and therefore correct average power consumption values. For example, we can easily distinguish $P(x_1|C_1)$, $P(x_1|C_2)$, $P(x_1|C_6)$, $P(x_1|C_7)$, $P(x_1|C_8)$ from each other in Fig. 1 (d). And the distributions of $P(x_2|C_3)$, $P(x_2|C_4)$, $P(x_2|C_5)$ are different in Fig. 1 (e). The distributions in

Fig. 1 (d) and (e) complement each other in the sense that similar distributions of those classes in Fig. 1 (d) are distinctive in Fig. 1 (e). In practice, multiple features need to work in concert to distinguish all classes. A number of options for feature extraction are experimented for circuit power estimation.

(1) Direct encoding of primary inputs/outputs

For a specific pin of primary inputs/outputs, let the feature x be the floating-point value 0.0, 1.0, 2.0, 3.0 when the pin undergoes 0→0, 1→1, 0→1, 1→0 transition, respectively. In the case of leakage power estimation, the only valid encoded values will be 0.0, 1.0. x can be further optionally scaled to a target value range (e.g. between 0.0 and 1.0). Depending on circuit topologies, exclusive use of this feature extraction option is found to be only effective for circuits with small number of primary inputs.

(2) Hand-crafted features

This is the method of encoding circuit specific features by analyzing circuit topology, identifying major power consumption components, examining clustered power classes with respect to state, transition, circuit topology, functionality and symmetry. It is a very effective way of finding good features for circuits with regular structures. Examples are the features discussed in section III for the circuit mux8.

(3) Statistical discriminant analysis

Without loss of generality, let's choose the encoding method described in section V-(1) for a selected primary input/output. Let n be the total number of sample SPICE power data points of a circuit, each with its associated power value (leakage power or switching energy) and its associated state/transition of primary inputs/outputs of the circuits. By choosing m of the primary inputs/outputs, and encoding each of the data points, we have an $n \times m$ matrix $\mathbf{Y}$. We can consider $\mathbf{Y}$ represent n data points in m-dimensional space. Assuming we would like to select k features as the inputs for the neural network, where $k \le m$. The original data matrix $\mathbf{Y}$ from m-dimensional space needs to be further transformed into k-dimensional space. This can be done by $\mathbf{Z}=\mathbf{YW}$, where $\mathbf{W}$ is the $m \times k$ transformation matrix. $\mathbf{W}$ needs to be selected in such a way that the classification accuracy of the neural network is maximized.

Good and effective features for neural networks need to have large variations among different sample points. For classification problems, features maximizing the variations between the clusters (classes) and at the same time minimizing the variations within clusters are desired. Selection of such features is studied in [13,14] from the perspective of statistical discriminant analysis.

Let $\mathbf{M}$ be the $1 \times m$ vector representing the mean of the n data points described by $\mathbf{Y}$. Further cluster the SPICE data points with respected to their power values into c clusters, as described in section III. Let n_i be the number of data points in cluster c_i, and encoding the selected m primary inputs/outputs accordingly, we have a number of $n_i \times m$ matrices $\mathbf{Y_i}$, where $i = 1, 2, \ldots, c$. Let $\mathbf{M_i}$ be the $1 \times m$ vector, representing the mean of n_i data points described by $\mathbf{Y_i}$ for cluster i. Let $\mathbf{S_w}$ be the $m \times m$ within-class scatter matrix defined by,

$$S_w = \sum_{i=1}^{c} (Y_i - M_i)^t (Y_i - M_i) \qquad (2)$$

The subtraction of matrix by vector in (2) is done by subtracting every row of the matrix $\mathbf{Y_i}$ by the same vector $\mathbf{M_i}$. Let $\mathbf{S_b}$ be the $m \times m$ between-class scatter matrix defined by,

$$S_b = \sum_{i=1}^{c} (M_i - M)^t (M_i - M) \qquad (3)$$

It is shown in [14] that the columns of $\mathbf{W}$, which simultaneously maximizing between-class variation and minimizing within-class variation, consist of the k eigenvectors associated with the k largest eigenvalues of the matrix $S_w^{-1}S_b$. It has been further shown in [15] that the numerical solution of the eigensystem of $S_w^{-1}S_b$ is stable, considering the fact that both $\mathbf{S_w}$ and $\mathbf{S_b}$ are symmetric, by using the following method: first perform spectral factorization $S_w = H\Lambda H^t$, define $X = \left(H\Lambda^{-\frac{1}{2}}\right)^t S_b H\Lambda^{-\frac{1}{2}}$, followed by a second spectral factorization $X = U\Sigma U^t$, further define $\nabla = H\Lambda^{-\frac{1}{2}}U$, and $S_w^{-1}S_b = \nabla\Sigma\nabla^{-1}$. In other words, Σ, ∇ consists of the eigenvalues, eigenvectors of $S_w^{-1}S_b$, respectively. Computing the transformation matrix $\mathbf{W}$ using this method provides another good option of selecting effective features for neural networks used for circuit power estimation.

VI. Experimental Results

The circuit power estimation technique described in previous sections has been applied to the entire library of several hundred static/dynamic circuits used for the design of a microprocessor and several SoC products. The neural network simulator SNNS [16] has been used to construct, train, validate, and generate C code of neural networks for leakage power and switching energy estimation of all the circuit topologies in the library. The C code of these neural networks has been further incorporated into a C++ circuit power library that models the internal power consumption of all circuits. Along with the gate switching activities gathered from logic simulation, this C++ circuit power library has been routinely used by a gate-level power estimation methodology to produce full-chip cycle-by-cycle power consumption profiles of hundreds of thousands of clock cycles for the microprocessor and SoC products under development. In this section, we select representative circuit topologies, and discuss the modeling accuracy of transition-dependent circuit internal switching energy compared to SPICE data. The number of primary inputs of these circuits range from 3 (decode38) to 64 (cmp32), covering the range of the transition space size from 56 to 3.4×10^{38}. For the same circuit, the modeling method of state-dependent leakage power is exactly the same, but it is a much easier problem, because the size of the state space is much smaller than the size of the transition space.

The power modeling accuracy will depend on the number of clusters (classes) over the entire power spectrum and classification accuracy. The most prominent advantage of neural networks is its capability of predicting unforeseen scenarios of the entire state/transition space. In cases where the complete enumeration of state/transition space is not possible, 60% of the data set is used to train the neural network, while the remaining 40% of the data is used for validation. Special effort has been made to ensure that the classification accuracy between the

mutually exclusive training and validation data closely tracks each other. The method of using a single mean value for all transitions is used as the benchmark for comparison. This benchmark reflects the variation of the switching energy with respect to different transitions. The comparison results are shown in Fig. 4 and 5. It can be seen that the characteristics of the benchmark curves are highly circuit topology dependent and the variation around the mean is high. For circuits with a small number of primary inputs, no coherent pattern can be observed. When the number of primary inputs increases, the benchmark curve approaches normal distribution. On the other hand, the estimation errors using neural networks always observe distributions similar to normal distribution, with much smaller variation compared to the benchmark curves. The estimation error decreases when the number of clusters increases and the more complicated neural networks are used, provided statistical difference of the power data set between different clusters exist and appropriate features are selected. Most of the computing time required for this approach has been spent on using SPICE (or SPICE-like fast circuit simulator) to collect circuit power consumption data sets that represent statistical behaviors of circuit power consumption. The time used for training and validating neural networks is negligible by comparison.

VII. Conclusions

A circuit power estimation method using Bayesian inference and neural networks has been proposed. Based on statistical distribution of circuit leakage power and switching energy, the entire state and transition space of a specific circuit are classified using neural networks into a limited few classes that represent different power consumption average values. This technique enables efficient table-lookup of circuit power of the entire state and transition space. This method involves gathering statistical information, clustering power consumption values, feature extraction for neural networks of circuit leakage and switching energy, construction, training and validation of neural networks, and table-lookup of circuit leakage and switching power using the validated neural networks. Experimental results on a wide range of circuit topologies demonstrated the feasibility of using the proposed method for estimating state-dependent leakage power and transition-dependent switching energy of library circuits in a cycle-by-cycle power estimation methodology. Although the focus of this study has been on power consumption modeling of library circuits, the proposed method and its variations may also be feasible for the power estimation of functional blocks, or even high-level power estimation.

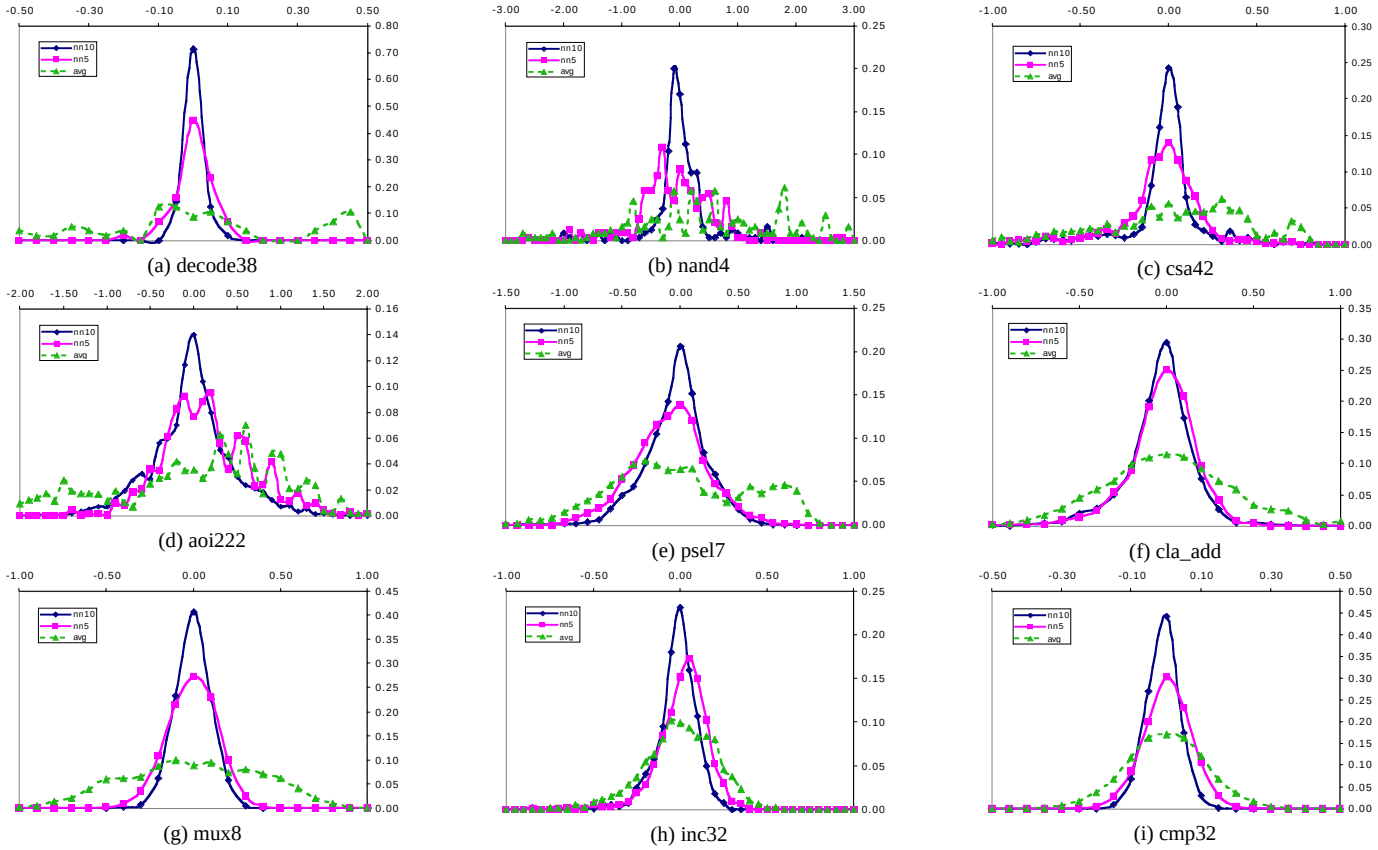

Fig 4. Probability distribution of estimation error compared to SPICE for different circuits. y-axis is the probability. x-axis is the estimation error. Let *avgval* be the mean of all the SPICE simulated values for a circuit. Let $estval_i$, $actval_i$ be the estimated value of switching energy using the circuit power model for a specific transition t_i, the SPICE simulated switching energy for the same transition t_i, respectively. Estimation error of neural networks for a transition t_i is defined as $nnerr_i = (estval_i - actval_i) / avgval$. "nn10", "nn5" denotes the estimation error for neural networks with 10, 5 outputs, respectively. "avg" denotes the estimation error, defined as $avgerr_i = (avgval - actval_i) / avgval$, if the mean is used as the transition-independent estimated power value. The curves in figures include both training and validation data.

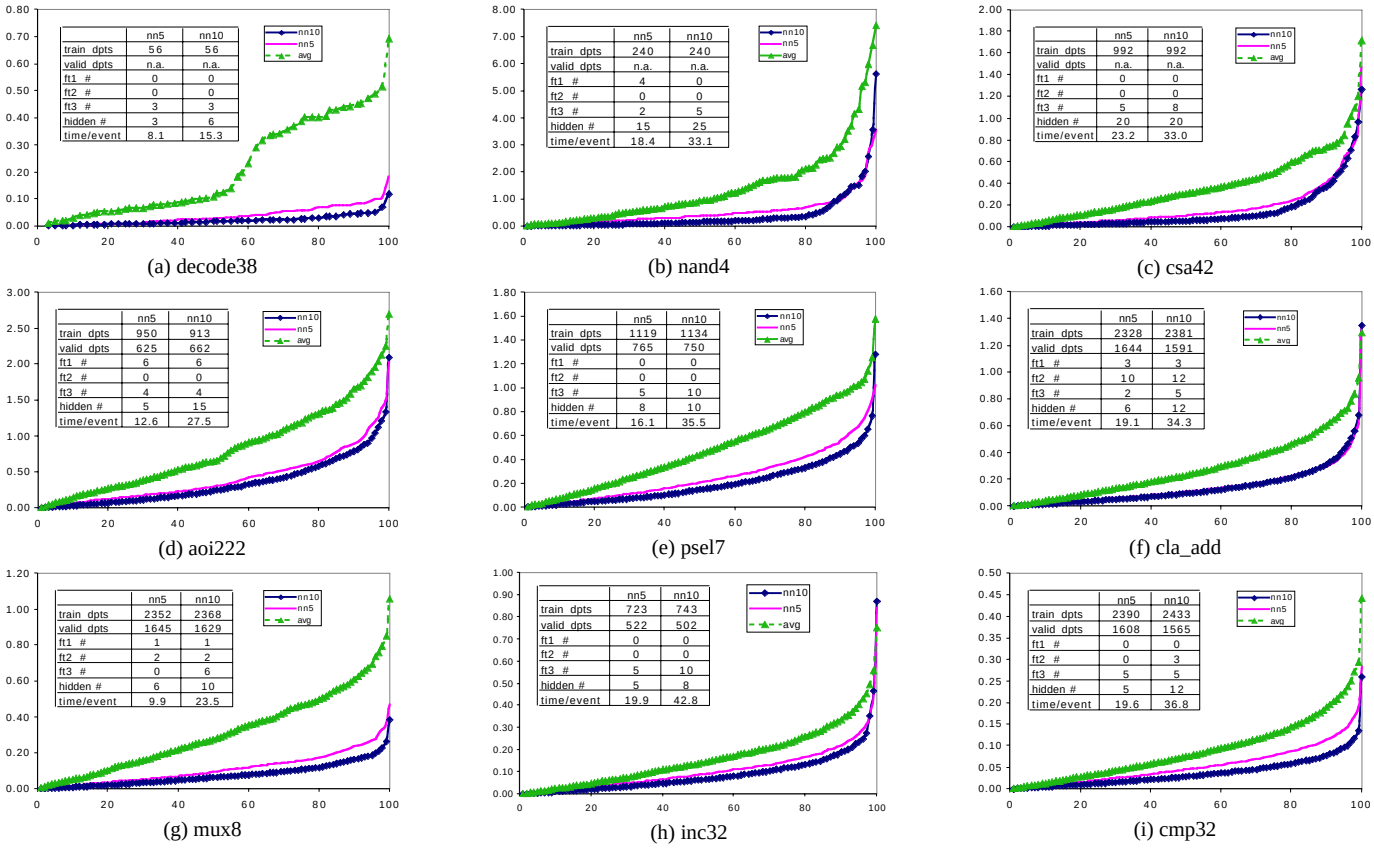

Fig 5. Cumulative estimation error compared to SPICE for different circuits and metrics of C++ circuit power neural network models. x-axis is the percentage of total data points. y-axis is the absolute value of estimation error. A point (x,y) in graph is interpreted as absolute value of estimation error is less than y for x% of cases. The definitions of estimation error, "nn10", "nn5", and "avg" are the same as those described in Fig. 4. "train dpts", "valid dpts" denotes the number of training and validation data points used to train/validate the neural network, respectively. Training and validation data sets are mutually exclusive. An "n.a." entry for "valid dpts" indicates the training data set is completely enumerated. The curves in figures include both training and validation data. "ft1 #", "ft2 #", "ft3 #" denotes the number of features constructed by methods described in section V-(1), V-(2), V-(3), respectively. "hidden #" denotes the number of hidden units of a neural network. "time/event" is the average execution time in μs for evaluating the switching energy of a single switching event of a specific circuit on a Sun Ultra 60/360 workstation for the C++ circuit power library compiled by gcc-3.0.x. This time period includes feature extraction, classification using neural networks, and table lookup of average switching energy value.

Acknowledgements

The author would like to thank helpful discussions with Hector Sanchez, Kamal Khouri, and Lyren Brown on power related issues during the period of this study.

References

[1] P. Landman, "High-level Power Estimation," in *Proc ISLPED*, pp. 29-35, 1996.

[2] S. Gupta and F. Najm, "Power Modeling for High-Level Power Estimation," *IEEE Trans VLSI Systems*, pp. 18-29, Feb 2000.

[3] F. Najm, "A Survey of Power Estimation Techniques in VLSI Circuits," *IEEE Trans VLSI Systems*, pp. 446-455, Dec. 1994.

[4] Y.K. Chen, D. Bearden, and K. Suryadevara, "Application-based, Transistor-level Full-chip Power Analysis for 700 MHz PowerPC Microprocessor," in *Proc ICCD*, pp. 215-220, 2000.

[5] *Library Compiler User Guide*, Synopsys Inc, May 2000.

[6] R. Burch, et al. "A Monte Carlo Approach for Power Estimation," *IEEE Trans VLSI Systems*, pp. 63-71, March 1993.

[7] C.S. Ding, Q. Wu, C.T. Hsieh and M. Pedram, "Stratified Random Sampling for Power Estimation," *IEEE Trans. CAD*, Vol. 17, No. 6, pp. 465-471, June 1998.

[8] G. Wang and R. Gopisetty, "Efficient Generation of Timing and Power Polynomial Models from Lookup Tables for SoC Designs," in *Proc 12th IEEE International ASIC/SOC Conference*, pp. 216-220, 1999.

[9] S. Bhanja and N. Ranganathan, "Dependency Preserving Probabilistic Modeling of Switching Activity Using Bayesian Networks," in *Proc* DAC, pp. 209-214, 2001.

[10] R. Marculescu, D. Marculescu, and M. Pedram, "Probabilistic Modeling of Dependencies During Switching Activity Analysis," *IEEE Trans. CAD*, Vol. 17, No. 2, pp. 73-83, February 1998.

[11] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[12] B.D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, 1996.

[13] R.A. Fisher, "The Statistical Utilization of Multiple Measurements," *Annals of Eugenics*, Vol. 8, pp. 376-386, 1938.

[14] S.S. Wilks, *Mathematical Statistics*, New York:Wiley, 1964.

[15] D.L. Swets and J. Weng, "Using Discriminant Eigenfeatures for Image Retrieval," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 18, No. 8, pp. 831-836, August 1996.

[16] Stuttgart Neural Network Simulator User Manual, version 4.2, 2000.