Zejia LYU, Jizhong SHEN, Xi CHEN, 2025. Algorithm and evaluation of generating pseudo-datasets for integrated circuit power analysis. *Frontiers of Information Technology & Electronic Engineering*, 26(9):1596-1608. https://doi.org/10.1631/FITEE.2400677

Algorithm and evaluation of generating pseudo-datasets for integrated circuit power analysis

Key words: Graph computation; Electronic design automation (EDA); Pseudo-dataset; Average power analysis

Corresponding author: Jizhong SHEN
E-mail: jzshen@zju.edu.cn
ORCID: https://orcid.org/0000-0002-9031-2379

Motivation

1. Power analysis plays a pivotal role in the design of large-scale digital integrated circuits (ICs), where accurate and efficient power evaluation during the design phase can substantially improve the chip performance.

2. Despite the advancements in power analysis techniques, the requirement for extensive circuit data and diverse topological scenarios remains a significant challenge.

3. In response to the limitations of existing datasets, to support the needs of machine learning (ML)-driven average power analysis methods, we propose a novel pseudo-circuit dataset generation algorithm specifically tailored for power analysis.

Main idea

1. Random directed acyclic graphs (DAGs) are created, followed by mapping DAG nodes to gate-level nodes to form combinational circuits.

2. By integrating register units, these combinational circuits are converted into sequential circuits, which are subsequently optimized using the Synopsys design compiler.

Method

1. We propose an efficient and rapid algorithm for generating random DAG topologies and then generating random combinational circuit netlists.

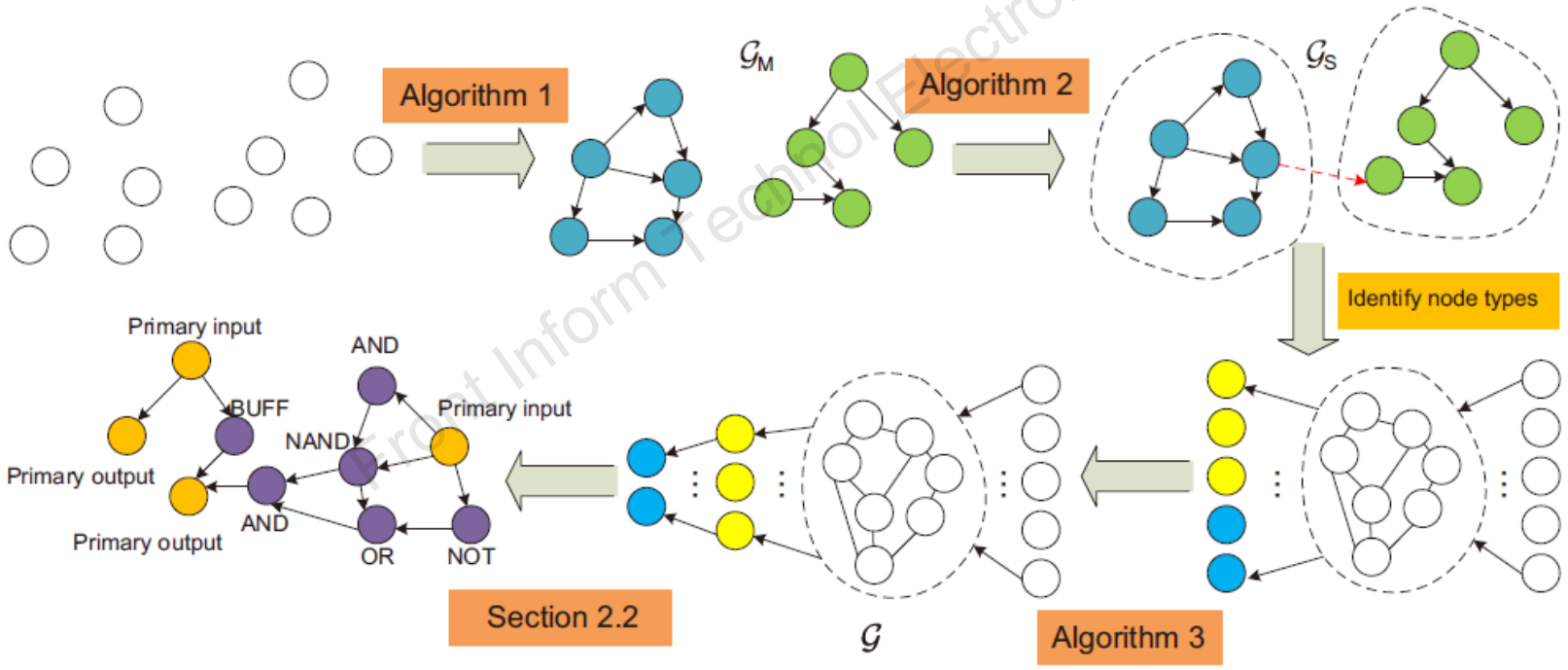

Fig. 2 Random combinational circuit netlist generation. Algorithm 1 processes the nodes into different connected components, Algorithm 2 transforms multi-connected components into a single-connected component, and Algorithm 3 adjusts the number of identified output pins. Finally, the generated DAG topology is converted into the combinational circuit netlist in Section 2.2

Method (Cont'd)

1.1 We introduce the random multi-connected component DAG generation method in Algorithm 1, which efficiently establishes the initial connections within *G*.

Algorithm 1 Random multi-connected component DAG generation method

Require: Graph $\mathcal{G}$ defined only by nodes without edges
Ensure: Multi-connected DAG graph $\mathcal{G}_M$

$\mathcal{S} \leftarrow \mathcal{W} + \mathcal{X}$
for all $o \in \mathcal{O}'$ **do**
  Randomly select γ nodes from $\mathcal{S}$ as parent nodes for o, forming the set $\mathcal{P}$
  for all $p \in \mathcal{P}$ **do**
    Add edge $\langle p, o \rangle$ to $\mathcal{G}$
    if $\mathcal{G}$ does not conform to DAG structure **then**
      Remove edge $\langle p, o \rangle$
    end if
    Add p to $\mathcal{S}$
  end for
end for

Method (Cont'd)

1.2 Algorithm 2 enhances the graph structure by examining and connecting separate components, ensuring that the graph maintains weak connectivity and forms a single-connected component.

Algorithm 2 Single-connected component assurance

Require: $\mathcal{G}_M$

Ensure: $\mathcal{G}_S$

1: $\mathcal{G}_S = \mathcal{G}_M$
2: **while** $\mathcal{G}_S$ is not weakly connected **do**
3: List all connected components of $\mathcal{G}_S$
4: Select different nodes from two different components
5: Connect these two nodes
6: Update $\mathcal{G}_S$
7: **end while**

Method (Cont'd)

1.3 By post-processing with Algorithm 3, the resultant computational graph forms a single-connected DAG, which is suitable for implementation in combinational circuits.

Algorithm 3 Adjustment of the primary output nodes

Require: $\mathcal{G}_S$ and the expected output node number n
Ensure: Computational graph $\mathcal{G}$ with adjusted output

Let $\mathcal{G} = \mathcal{G}_S$
while true **do**
  $\mathcal{O}_S \leftarrow \mathcal{G}$
  $\mathcal{O}_S^+, \mathcal{O}_S^- \leftarrow \mathcal{O}_S$
  for all $\sigma \in \mathcal{O}_S^+$ **do**
    Select nodes from $\mathcal{O}_S^-$ as parent $\mathcal{P}_S$ for σ
    for $p \in \mathcal{P}_S$ **do**
      Add edge $\langle p, \sigma \rangle$ to $\mathcal{G}$
    end for
  end for
  if the number of output nodes in $\mathcal{G}$ approximates n **then**
    Break loop
  end if
end while

Method (Cont'd)

2. The transformation from a combinational circuit netlist to a sequential circuit netlist can be facilitated by introducing registers. This method involves selecting a specific number of primary inputs and outputs from the existing gate-level combinational circuit and connecting them using predefined gate-level registers. This process effectively transforms the DAG combinational circuit into a directed cycle graph (DCG) structure pseudo-sequential circuit netlist.

Major results

Pseudo-power datasets and comparison with state-of-the-art dataset methods

Table 1 Overview of pseudo-power datasets

Type	Metric	Number of input nodes	Number of output nodes	Number of wires	Number of logical gates	Number of registers
Combinational circuit	Mean	49	58	3311	3370	
	Min	22	35	887	939	
	Max	79	83	5638	5715	
Sequential circuit	Mean	254	256	4769	4918	107
	Min	90	98	686	1104	37
	Max	436	440	4950	7429	185

Table 2 Comparison between the proposed power consumption dataset and the ML EDA dataset of application-specific integrated circuit (ASIC) and field programmable gate array (FPGA) in existing studies

Work	Number of cases	Number of sources	Platform & level	Usage
CircuitNet (Chai et al., 2022)	20 000	6	ASIC PR design	DRC; IR drop
OpenABC-D (Chowdhury et al., 2021)	870 000	29	ASIC RTL	Synthesis
HLSDataset (Wei et al., 2023)	19 000	34	FPGA HLS	Power; timing
Spector (Gautier et al., 2016)	8000	9	FPGA HLS	NA
Proposed	Unlimited	NA	ASIC gate-level Verilog	Power

NA: not available; PR: placement & routing; HLS: high-level synthesis; DRC: design rule checking

Major results (Cont'd)

Power result

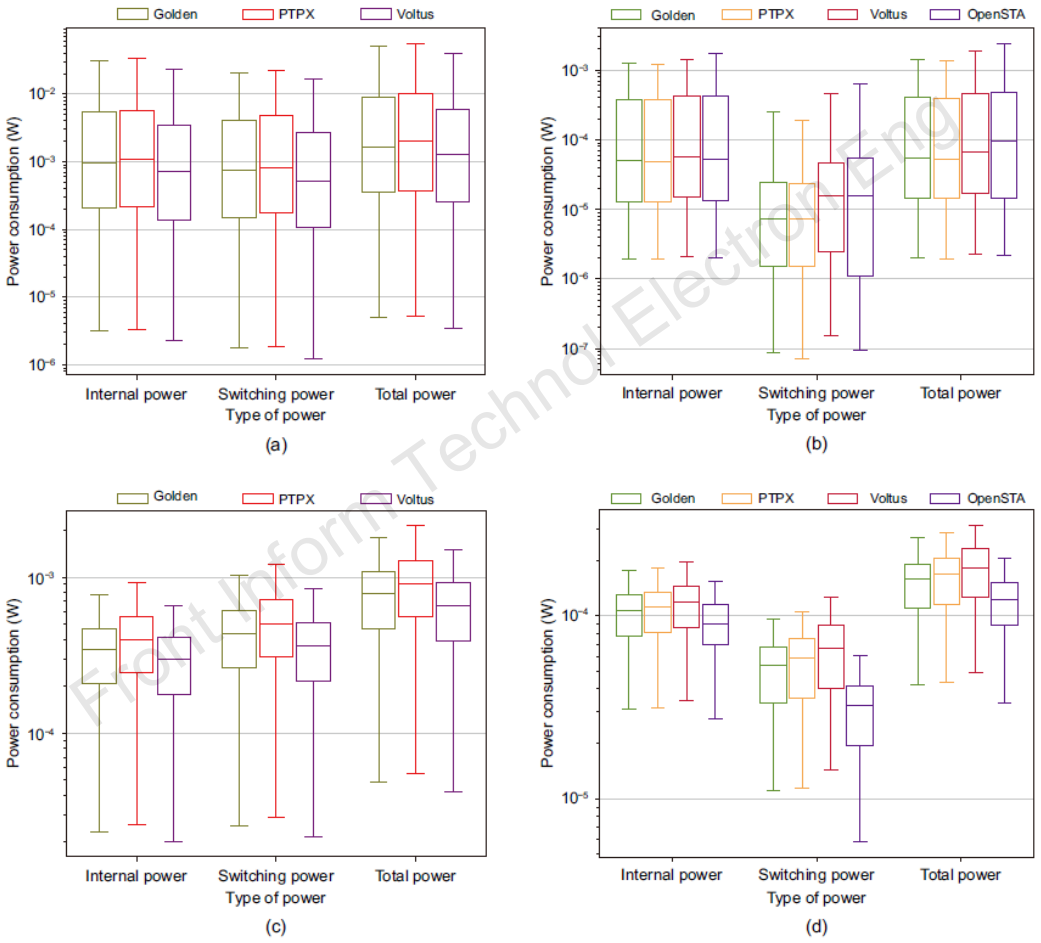

Fig. 4 Results of the three power consumptions for all samples shown on the left for combinational circuits and on the right for sequential circuits: (a) benchmark combinational circuit power consumption; (b) benchmark sequential circuit power consumption; (c) proposed combinational circuit power consumption; (d) proposed sequential circuit power consumption. References to color refer to the online version of this figure

Major results (Cont'd)

Simulations on the topological complexities

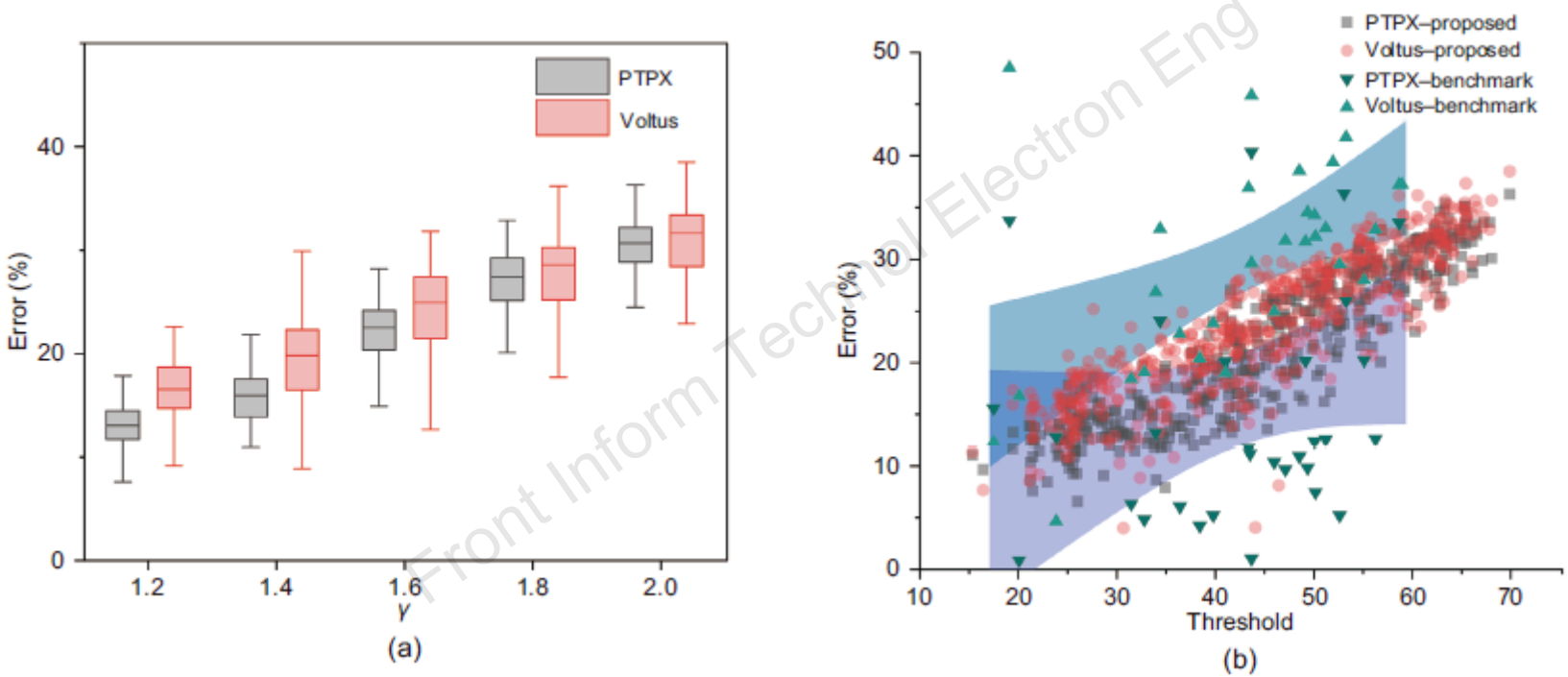

Fig. 5 Simulations on the topological complexities of the generated combinational circuits, where the vertical coordinates are all the total power consumption errors: (a) effect of γ value on the power consumption error; (b) threshold measures, where the upper blue region is the 95% confidence interval of the PTPX error for benchmark and the lower purple region is the 95% confidence interval of the Voltus error

Conclusions

1. Unlike traditional approaches, the proposed method uses graph topology computation to efficiently generate random DAG structures.

2. These DAGs are transformed into pseudo-combinatorial circuit netlists, with registers added to convert them into pseudo-sequential circuit netlists.

3. We manage the complexity of these circuits by setting hyperparameters. By examining complexity and comparing it with that of the benchmark datasets, the results demonstrate and validate the effectiveness of the proposed dataset.

Zejia LYU received the B.S. degree in electronic science and technology from Zhejiang University, Hangzhou, China, in 2019, where he is currently pursuing the Ph.D. degree in the College of Information Science and Electronic Engineering. His current research interests include brain-inspired computing, EDA, and computer architecture.

Jizhong SHEN received the Ph.D. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2001. He is currently a full professor in the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. He has authored or coauthored more than 190 refereed technical articles on digital circuit design, low-power design, and brain–computer interface. His current research interests include digital integrated circuit, EDA, and brain–computer interface.