Learning Monte Carlo Data for Circuit Path Length

Namal A. Senanayake, A. Beg, and Withana C. Prasad

Abstract—This paper analyzes the patterns of the Monte Carlo data for a large number of variables and minterms, in order to characterize the circuit path length behavior. We propose models that are determined by training process of shortest path length derived from a wide range of binary decision diagram (BDD) simulations. The creation of the model was done use of feed forward neural network (NN) modeling methodology. Experimental results for ISCAS benchmark circuits show an RMS error of 0.102 for the shortest path length complexity estimation predicted by the NN model (NNM). Use of such a model can help reduce the time complexity of very large scale integrated (VLSI) circuitries and related computer-aided design (CAD) tools that use BDDs.

Keywords—Monte Carlo data, Binary decision diagrams, Neural network modeling, Shortest path length estimation.

I. INTRODUCTION

THE increasing complexity of modern VLSI circuitry is only manageable through advanced CAD systems that allow efficient handling of BFs (BFs) [1]. One of the most important functions of CAD tools is to provide robust and efficient data structures to represent BFs as well as fast algorithms to manipulate these data structures. During the last two decades, BDDs have gained popularity as the data structures in solving most of the combinational problems which arise in synthesis and verification of digital systems.

BDD in general is direct acyclic graph representations of BFs. BDDs were proposed by Akers [2] and were further generalized by Bryant [3]. The success of BDDs has attracted many researchers in the area of design, synthesis and verification of VLSI circuits. Evaluation of the time complexity of a BF can be performed by employing its BDD [3].

Fast evaluation time is a key step in many applications such as logic simulation, testing evaluation process of logic circuits [4], [5]. As the circuit sizes continue to grow, the need for fast evaluation becomes even more significant. The evaluation time is not directly related to the number of nodes in a BDD, but it is proportional to the path length of the BDD. Therefore, minimization of the path length can improve the overall performance of the circuit implementing BFs. This will eventually increase the efficiency of the final implementation [6], [7]. Numerous research works have been done to analyze the behavior of path related objective functions [6]-[10]. Most of the proposed methods are based on either static variable ordering [11]-[14] or dynamic variable ordering techniques [15], [16]. The minimization of the APL leads to circuits with smaller depth of paths from the root to the terminal node of the BDD. The resulting circuit will be optimized for speed on one hand, and on the other hand the number of very long paths in the BDD will be reduced [17]. The minimization of Average Path Length (APL) is of great importance in embedded systems, real time operating system applications [18], [19]. Minimization of Longest Path Length (LPL) [6] and Shortest Path Length (SPL) in BDDs is motivated by the synthesis of digital circuits in order to optimize their delays, which is a very important issue for Pass Transistor Logic (PTL) [20], [21].

In all these path length minimizations, we need to create the whole BDD representing the BF with the best possible variable ordering method. Building the whole BDD may lead to some complexity in the design process in terms of the time required to implement, verify and test the design. So it will be useful to have an estimation of the BDD complexity prior to make decisions on the feasibility of the design. Many research works have been published on the estimation of combinational and sequential circuits [22]-[24].

NNs learn from experience or some known examples. The learning process results in a set of weights that tend to minimize the errors between the NNM and the actual examples. NN are useful at pattern recognition, generalization, and trend prediction. Over the decade NN has been used to provide solution to difficult NP-complete optimization problems [25]. The measure of efficiency of the circuits has been addressed in relation with the area of circuit implementation, where the complexity of BFs is analyzed in terms of their implementation using different kind of circuits, from those with simple sum-of-product (SOP) to NNs. In recent times, some research work has been done on BF complexity analysis using NN learning process. [26], [27]. The main idea of this paper is to extend the work done by the same authors to demonstrate the capabilities of a NN methodology in effectively modeling the behavior of path length properties. Previously this methodology addressed the

Namal A. Senanayake is with the Monash University Sunway Campus, Malaysia since 2002 (phone: +603-55146249; fax: +603-55146207; e-mail: senanayake.namal@eng.monash.edu.my).

Azam Beg is with College of Information Technology, United Arab Emirates University, Al-Ain, UAE (e-mail: abeg@uaeu.ac.ae).

W. C. Prasad is with College of Information Technology, United Arab Emirates University, Al-Ain, UAE, on leave from Multimedia University, Malaysia (e-mail: prasadc@uaeu.ac.ae).

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:3, 2008

LPL and APL of BDD. Here we use an NNM to predict the SPL complexity with Monte Carlo BDD simulation.

In second section of this paper, we review the previous work done by the same authors on the estimation of path length properties. The proposed NNM for the estimation of SPL complexity is explained in the third section. Section four provides the ISCAS benchmark validation for the NNM. Finally we conclude our paper with our future developments in the same area.

II. PREVIOUS WORK DONE ON PATH LENGTH COMPLEXITY ESTIMATION

We used an NN software package called Brain-Maker version 3.75 [28] to model the APL and LPL behavior [29]. Brain Maker's feed-forward back-propagation NNs were fully connected, meaning all inputs were connected to all hidden neurons, and all hidden neurons were connected to the outputs. Our experiments involved different number of neurons in the single hidden layer. We used 90% of the data sets as the *training set* and the other 10% as the *validation set*. During training, only the training set was presented to the NNs, and not the validation set. We had acquired a total of 10,528 data sets (also called *facts*) by running BF simulations [29]. A total of 72 different configurations of NNM were used to collect the data on NNM learn-ability. A given NNM was considered to be sufficiently trained when it had learnt 97.5% of the training facts. For our NNMs, the raw data (using no transformation) provided APL and LPL average training accuracy of 90.8%, 89.3% and average validation accuracy of 90% and 90.5%, respectively. The Fig. 1 illustrates the comparison of APL and LPL complexity for 10 variables from simulations and NNM predictions.
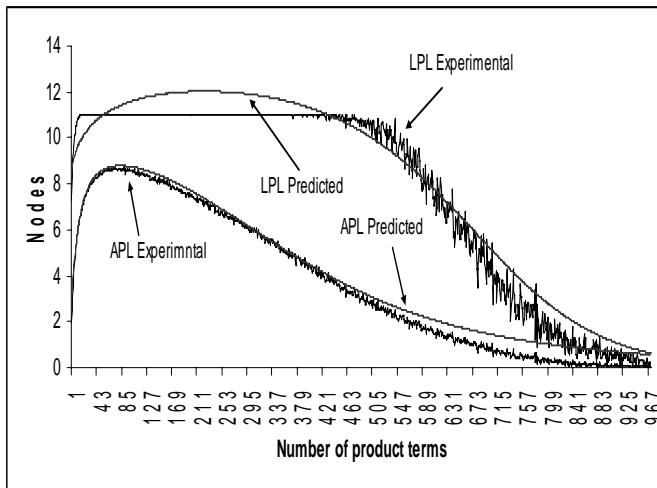

Fig. 1 Comparison of APL and LPL complexity for 10 variables from simulations and NNM predictions

III. DATA ACQUISITION AND PRE-PROCESSING

For each variable count n between 1 and 14 inclusive and for each term count between 1 and 2^n-1, 100 SOP terms were randomly generated and the Colorado university Decision Diagram (CUDD) package [30] was used to determine the SPL in terms of nodes. This process was repeated until the average size of the SPL complexities (i.e. number of nodes) became 1. Then the graphs for both the complexities were plotted against the product term count for number of variables 1 to 14.

A. NN Training Setup and Testing Accuracy

The NN-modeling software package Brain Maker has been used here to create and test the NNMs, as mentioned earlier. The configuration and training statistics for SPL is given in Table I. It shows that our experiments involved different number of neurons in the single hidden layer.

TABLE I
CONFIGURATION AND RELATED STATISTICS FOR SPL COMPLEXITY NNM

Neurons in single hidden layer	Training epochs	Training time (hours)	Training accuracy %	Validation accuracy %
2	204	0.02	59	61
4	384	0.04	94	93
6	397	0.04	91	90
10	263	0.04	94	94
14	333	0.04	94	94
18	341	0.04	95	95
20	235	0.03	95	94
22	174	0.02	95	95
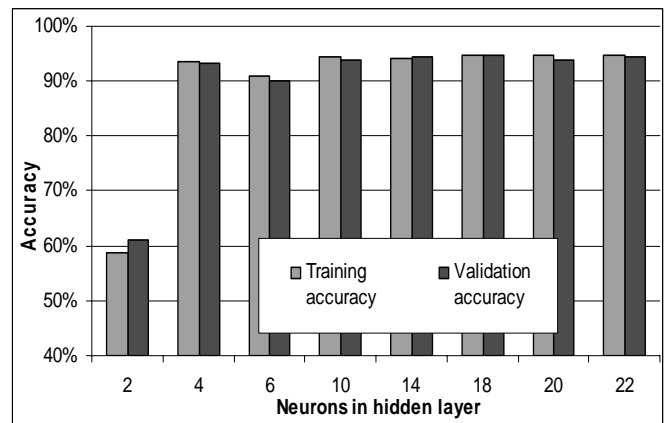
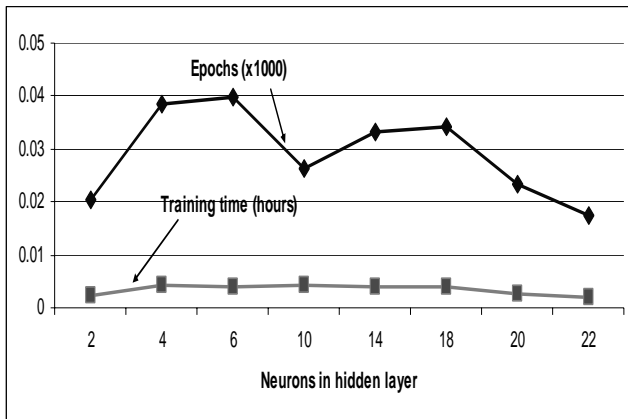Fig. 2 Training and validation accuracy for different number of neurons in the hidden layer for SPL

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:3, 2008

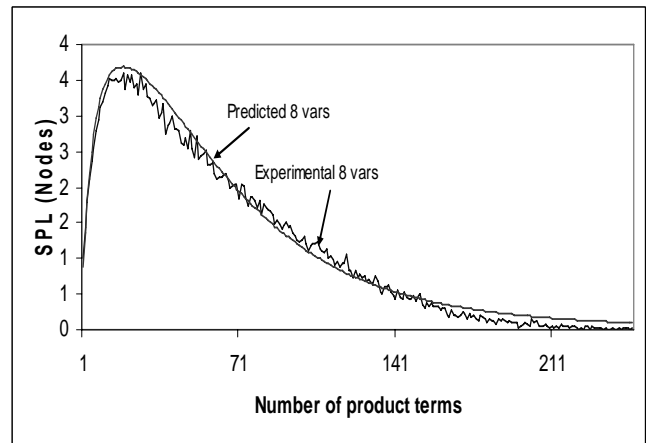Fig. 3 Training time as a function of hidden layer size

Fig. 4 Comparison of SPL experimental (blue) and predicted (red)
results for 8 variables
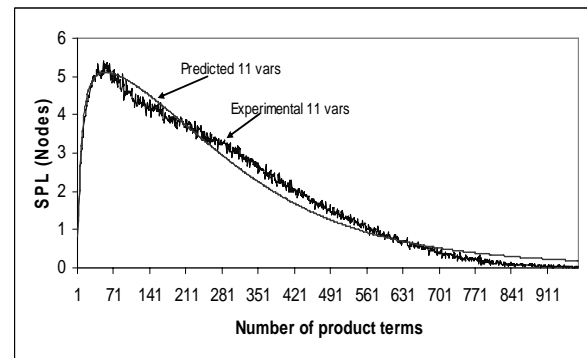
Fig. 5 Comparison of SPL experimental (blue) and predicted (red)
results for 11 variables

For our NNMs, the raw data for SPL provided an average training accuracy of 89.6% and average validation accuracy of 89.5%. Fig. 2 illustrates the training and validation accuracy as a function of neuron count in the (single) hidden layer for SPL. As expected, we needed fewer training epochs as the number of neurons in the hidden layer was increased; this is indicated by the trend-line in Fig. 3. Another point worth mentioning is that each NN configuration was trained multiple times and the best training statistics for every configuration were collected to alleviate the issue of local minima. Any increase in hidden-layer neuron count beyond four had a marginal improvement in the model accuracy. The closeness of training and validation accuracies validate the performance of our NNMs. Table I shows Configuration and related statistics for SPL-complexity NNMs.

B. NN Modeling Results and Analysis

We used an arbitrary set of values for number-of-variables and number of product terms and used the NNM to predict the SPL complexities in the form of nodes (complexity).

Figs. 4 and 5 illustrate the comparison for experimental results and NNM predictions of SPL complexities for 8 and 11 variables respectively. It can be inferred that the NNM result provides a very good approximation of the path related objective function complexity.

The NNM could also be used for prediction of path length properties beyond 14 variables as the NNMs are somewhat capable of extrapolation [28].

IV. NN MODEL VALIDATION

Table II Illustrates the ISCAS benchmark circuit [31] validation results for simulation using CUDD package and the proposed NNM for SPL complexity estimation. The ISCAS benchmarks are sets of multi-input compound Boolean expressions, because the randomly generated BFs used for the experiments were single output SOP expressions and the benchmark functions were split into multiple single-output expressions, and then expanded directly to SOP term. Each ISCAS benchmark produced a collection of SOP expressions. For each of these expressions, the node count was computed using the CUDD package [30]. For some benchmarks, lack of variation made the correlation meaningless. But, for the complete set of 426 circuits, the NNM was able to produce the match with the RMS error of 0.102 is very significant. It can be inferred from these results that the NNM is a better model on prediction of the SPL complexity if the input data range is known. Although the benchmark circuits considered had up to 94 inputs, mostly those benchmarks consisted of product terms of 1-14 variables. The circuits for all outputs were measured. It was observed that the term-variable count combinations were almost all to the left of the roll off of the graph, and thus still in region of logarithmic complexity. So, empirically the most important part of the model is the

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:3, 2008

logarithmic rise, and it was this part that has been validly tested by the benchmark circuit analysis. It is obvious that importance of a full-scale match of the curves will be more difficult to justify because of the lack of sample minterms that can be extracted from the benchmarks.

TABLE II
ISCAS BENCHMARK CIRCUIT VALIDATION

Circuit name	Number of input variables	Number of circuits	SPL complexity		
			Actual	NNM	Relative error
5xp1	7	10	16.890	17.464	0.034
alu4	14	8	30.270	32.031	0.058
apex7	48	55	72.410	65.865	-0.090
b1	3	4	4.490	2.863	-0.362
b12	15	9	22.940	24.165	0.053
b9	41	21	44.030	45.050	0.023
c8	28	17	35.760	32.071	-0.103
cc	21	18	27.460	24.721	-0.100
cht	47	36	22.880	26.287	0.149
clip	9	5	19.690	21.480	0.091
cmb	16	4	13.160	14.455	0.098
con1	6	2	4.060	3.952	-0.027
cu	14	11	10.260	10.383	0.012
decod	5	16	32.800	30.112	-0.082
inc	15	57	23.670	25.030	0.057
majority	5	1	2.050	1.882	-0.082
misex1	8	7	12.060	12.344	0.024
misex3	14	14	67.730	68.202	0.007
pcle	19	9	22.320	23.508	0.053
pm1	9	13	16.750	16.389	-0.022
sao2	10	4	13.020	13.946	0.071
sct	14	15	26.150	27.311	0.044
sqrt8	8	4	5.481	5.957	0.087
squar5	5	8	14.980	12.274	-0.181
ttt2	24	12	49.020	50.289	0.026
x2	10	7	23.260	21.701	-0.067
x4	94	59	151.550	152.267	0.005
Total circuits		426	RMS error		-0.0083

V. CONCLUSION

In this research work, we extended the work done by the authors in relation of NNMs with the path length properties, mainly shortest path length. The NNM was obtained through the training utilizing the experimental data for Monte Carlo BDD simulation data. The ISCAS benchmark validation with RMS errors of 0.102 has shown the accuracy of the training model. It also demonstrated that the NNMs were capable of providing useful clues about the complexity of the final circuit. Once NNMs had been developed, they could be used to conduct further experiments with different types of inputs,

in a fraction of time what a circuit simulator would take. Future work will be mainly concentrated on having wider range of variables to verify the full-scale match of the curves.

REFERENCES

[1] K. Priyank, "VLSI Logic Test, Validation and Verification, Properties & Applications of Binary Decision Diagrams", Lecture Notes, Department of Electrical and Computer Engineering University of Utah, Salt Lake City, UT 84112.

[2] S. B. Akers, "Binary Decision Diagram", IEEE Trans. Computers, Vol. 27, pp. 509-516, 1978.

[3] R. E. Bryant, "Graph–Based Algorithm for BF Manipulation", IEEE Trans. Computers, Vol. 35,, pp. 677-691, 1986.

[4] C. Scholl, R. Drechsler, and B. Becker, "Functional simulation using binary decision diagrams", Proc. Inter. Conf. of CAD, pp. 8-12, 1997.

[5] D. K. Pradhan, A. K. Singh, T. L Rajaprabhu, A. M. Jabir, "GASIM: A Fast Galois Filed Based Simulator for Functional Model", IEEE Proc. of HLDVT'05, pp. 135-142, 2005.

[6] S. Nagayama, and T. Sasao, "On the minimization of longest path length for decision diagrams", Proc Inter. Workshop on Logic and Synthesis (IWLS-2004),pp. 28-35, 2004.

[7] P.W. C. Prasad, M. Raseen, A. Assi, and S. M. N. A. Senanayake, "BDD Path Length Minimization based on Initial Variable Ordering", Journal of Computer Science, Science Publications, Vol. 1(4), pp. 521-529, 2005.

[8] Y. Liu, K. H. Wang, T. T. Hwang, and C. L. Liu, "Binary decision Diagrams with minimum expected path length", Proc. of DATE 01, pp. 708–712, 2001.

[9] R. Ebendt, S. Hoehne, W. Guenther, and R. Drechsler, "Minimization of the expected path length in BDDs based on local changes", Proc. of Asia and South Pacific Design Automation Conf. (ASP-DAC'2004), pp. 866-871, 2004.

[10] R. Ebendt, and R. Drechsler, "On the Exact Minimization of Path-Related Objective Functions for BDDs", Proc. of Inter. Conf. on Very Large Scale integration (IFIP VLSI-SOC), pp. 525-530, 2005.

[11] N. Drechsler, M. Hilgemeier, G. Fey, and R. Drechsler, "Disjoint Sum of Product Minimization by Evolutionary Algorithms", *Proc. of Applications of Evolutionary Computing,* Evo.Workshops, pp. 198-207, 2004.

[12] S. Nagayama, A. Mishchenko, T. Sasao, and J.T. Butler, "Minimization of average path length in BDDs by variable reordering", Proc. of Intl. Workshop on Logic and Synthesis, pp: 207-213, 2003.

[13] M. Fujita, H. Fujisawa, and N. Kawato, "Evaluation and Improvements of Boolean Comparison Method Based on Binary Decision Diagrams", Proc. of Inter. Conf. on Computer Aided Design (ICCAD), pp. 2-5, 1988.

[14] S. Malik, A. Wang, R. Brayton, and A. Sangiovanni-Vincentelli, "Logic Verification Using Binary Decision Diagrams in a Logic Synthesis Environment", Proc. of the Inter. Conf. on Computer Aided Design (ICCAD), pp. 6-9, 1988.

[15] R. Rudell, "Dynamic Variable Ordering for Ordered Binary Decision Diagrams", Proc. of the Inter. Conf. on Computer Aided Design (ICCAD), pp. 42-47, 1993.

[16] F. Somenzi, "Efficient manipulation of decision diagrams", Intl. Journal on. Software Tools for Technology. Transfer, (STTT), Vol. 3, pp. 171-181, 2001.

[17] G. Fey, J. Shi and R. Drechsler, "BDD Circuit Optimization for Path Delay Fault-Testability", Proc. of EUROMICRO Symposium on Digital System Design, pp. 168-172, 2004.

[18] A. Jain, M. Narayan, and A. Sangiovanni Vincentelli, "Formal Verification of combinational Circuits", Proc. of Inter. Conf. on VLSI Design, pp. 218-225, 1997.

[19] M. Lindgren, H. Hansson, and H. Thane, "Using Measurements to Derive the Worst-case Execution Time", Proc. of 7th Inter. Conf. on Real-Time Systems and Applications (RTCSA'00), pp. 15-22, 2000.

[20] V. Bertacco, S. Minato, P. Verplaetse, L. Benini, and G. De Micheli, "Decision Diagrams and Pass Transistor Logic Synthesis", Stanford University CSL Technical Report, No. CSL-TR-97-748, 1997.

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:3, 2008

[21] R. S. Shelar and S. S. Sapatnekar, "Recursive Bi-partitioning of BDD's for Performance Driven Synthesis of Pass Transistor Logic", Proc. of IEEE/ACM ICCAD, pp. 449-452, 2001.
[22] M. Nemani and F. N. Najm, "High-Level Power Estimation and the Area Complexity of BFs", Proc. of *IEEE Inter. Symposium on Low Power Electronics and Design*, pp. 329-334, 1996.
[23] N. Ramalingam, and S. Bhanja, "Causal Probabilistic Input Dependency Learning for Switching model in VLSI Circuits", Proc. of ACM Great Lakes Symposium on VLSI, pp. 112-115, 2005.
[24] P. E. Dunne, and W. van der Hoeke, "Representation and Complexity in Boolean Games", Proc. 9th European Conf. on Logics in Artificial Intelligence, LNCS 3229, Springer-Verlag, pp. 347-355, 2004.
[25] I. Parberry, Circuit Complexity and Neural Networks. MIT Press, 1994.
[26] L. Franco, M. Anthony, "On a generalization complexity measure for BFs", IEEE Conference on Neural Networks, Proc. of IEEE International Joint Conference on Neural Networks, pp. 973-978, 2004.
[27] L. Franco, "Role of function complexity and network size in the generalization ability of feedforward networks", Lecture Notes in Computer Science, v 3512, Computational Intelligence and Bioinspired Systems: *8th International Workshop on Artificial Neural Networks*, IWANN 2005, Proceedings, pp. 1-8, 2005.
[28] *BrainMaker – User's Guide and Reference Manual*, 7th ed., California Scientific Software Press, 1998.
[29] P.W.C. Prasad, A. Assi, and A. Beg, "Predicting the Complexity of Digital Circuits Using Neural Networks", WSEAS Transaction on Circuits and Systems, Vol. 5(6), pp. 813-820, 2006.
[30] F. Somenzi, "CUDD: CU Decision Diagram Package," *ftp://vlsi.colorado.edu/ pub/.*, 2003.
[31] S., Yang, "Logic synthesis and optimization benchmarks user guide version 3.0," Technical report, Microelectronic Centre of North Caroline, Research Triangle Park, NC, 1991.

Namal A. Senanayake (M'05–SM'06) is currently with Monash University Sunway Campus where he leads the research group Intelligent, Integrated and Interactive Systems (IIIS). He has been recently appointed as the chairman of IEEE Asia-Pacific Robotics & Automation Development Council – Malaysia Section. Prior to Monash, he has been working with three different universities; Chalmers University of Technology, Sweden, Johannes Kepler University of Linz, Austria and University of Peradeniya, Sri Lanka holding key academic and research positions. During his 18 years of research experience, he managed to publish over 72 publications in international conferences, journals and book chapters. He was one of the editors of three books published based on the research outcomes. He was the special session organizer for various international conferences, in particular IEEE conferences. He is one of the reviewers in IEEE publications and Elsevier publications.

He has initiated research with Sports Biomechanics Centre, National Sports Complex, in which his research team carried out special research projects of national interest. Having engaged in this area of research, Interactive Multilayer Sensorized Smart Floor has been developed under his leadership and currently in the process of patenting the device. Dr. Arosha is the leader of MoU between Monash and National Instruments. He carried out various special research projects under this MoU which are mainly targeting industrial needs.

Dr. Senanayake is a member of research committee of Monash and he is the student counselor of IEEE student branch at Monash

Azam Beg is with College of Information Technology, United Arab Emirates University, Al-Ain, UAE.

His research interest is on computer architecture, VLSI design and applying AI into circuit design and computer architecture.

Dr. Beg received his master's and Ph.D. from Mississippi State University, MS, USA.

Withana C. Prasad is with College of Information Technology, United Arab Emirates University, Al-Ain, UAE, on leave from Multimedia University, Malaysia.

His research interest is in VLSI design.

Dr. Prasad received his Master's from University of Ukrane and Ph.D. from Multimedia University, Malaysia.