Migrating Inputs: A Comparator Based Sorting

S. Ureeb and M. S. H. Khiyal, *Member*, *IACSIT*

***Abstract*—Sorting network problem seems to be a very simple, yet very complex but still maintains an attractive field for the researchers. The focus of the study is to review relevant work towards migration of smallest value to the uppermost horizontal output, while the largest to the lowest output signal in an iterative manner. The network depth, total number of comparators and the cost of the implementation of algorithm have the definite role in generalizing the solution set. This study illuminates the genuine and cogent arguments addressing the problem nodes in tune of migrating inputs, network efficiency, techniques previously studied optimization and cost constraints. The problem emerges as the input increases from 8<*n*<16 and becomes invincibly difficult when *n*>16 in parallel processing.The study also aims where the network inputs *n*=17.**

***Index Terms*—Comparators, cost, network optimization, parallel computing, sorting.**

I. Introduction

Sorting is a problem, which starts from a human's early age and lasts till he survives. Human brain has the capability to understand all the complex algorithms but question arises when these complex operations need to be implemented on a machine. Use of layers instead of neurons and parallel processing became difficult.

The simplest network consists of two inputs and needs only one comparator to sort between the two inputs. The graphical representation of migration of two network to upper and lower output signal shown in Fig. 1.

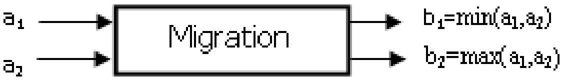


Fig. 1. Two input migrating model with two inputs and same number of outputs with a single operation of migration which will take place only when $a_2 < a_1$, otherwise no change will be produced on $a_2 \geq a_1$.

A comparator is a mapping:

$$
\begin{aligned}
&[i{:}\, j] : A^n \rightarrow A^n,\ \ i, j \in \{0, \dots, n\text{-}1\} \text{ with}\\
&[i{:}\, j](a)_i \ = \ \min(a_i, a_j)\\
&[i{:}\, j](a)_j \ = \ \max(a_i, a_j)\\
&[i{:}\, j](a)_k \ = \ a_k \ \text{ for all } k \ \text{ with } \ k \neq i,\ \ k \neq j\\
&\qquad \text{for all } a \in A^n.
\end{aligned}
$$

4-Input network as graphically represented in the Fig 2 with 4 inputs, 5 comparators and 5 output which are migrated inputs in ascending order as a result of comparison made. Here the network depth is 3. The comparison between (a_1, a_2) and (a_3, a_4) will be done simultaneously, however (a_1, a_3) comparator will wait for the results generated in first step and will be processed in parallel to (a_2, a_4), to generate the result in (a_2, a_3) after the step 3 and step 4.

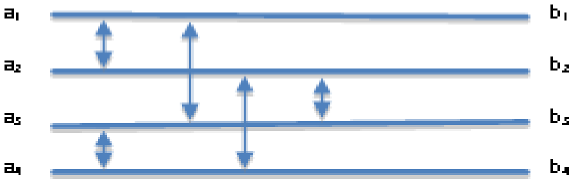


Fig. 2. Knuth [1] diagram may be defined as 4-Input Migrating Model, which results in a fully sorted array in increasing pattern with five(5) comparators and with the network depth of 3.

where $\{a_1,a_2,a_3,a_4\}$ comprises the input values, and $\{\mathbf{b_1}{\leq}b_2{\leq}b_3{\leq}b_4,\ b_1{\geq}\mathbf{b_2}{\leq}b_3{\leq}b_4, b_1{\geq}b_2{\geq}\mathbf{b_3}{\leq}b_4, b_1{\geq}b_2{\geq}b_3{\geq}\mathbf{b_4}\}$ is the output set. To find such minimal migrating network for 8≤*n*≤16 is still open end but beyond 16 it's a vast field.

Comparator based sorting is well versed for *n*≤8, but the field needs further improvements for networks with *n*≥8. All the researchers have made *n*=16 a classical problem and tried to give the minimal set of comparators with optimal depth and cost constraints. The most advanced sorting network needs 60 comparators to sort the network completely.

II. Literature Review

This study is focusing on almost all the techniques used to find the minimal amount of comparators, the cost factor analyzed by different researchers. Valsalan and Miikkulainen [2] used a two-step strategy to address the migrating networks by using symmetry in first step and then the evolutionary search. Much work is contributed by Juille [3] in the field of Evolutionary Sorting technique. END gives proved results for *n*=13, where the comparators were 45 and for *n*=16, the number of comparators are 60, which is unbeatable uptil now. Noticeable, issues have been addressed by Batcher [4] in the early age of the subject in bitonic Sorters and Odd/ Even sorting algorithm. He also addressed the depth and cost of the sorting networks. Keeping Batchers research principles has been made for further research in the field.

Genetic search algorithm also remained a popular topic in the field and noticeable work being proposed by Choi and Moon [1] which resulted in 60 comparators for the classical

Manuscript received April 22, 2014; revised August 20, 2014.

The authors are with Faculty of Computer Science, Preston University, Islamabad, Pakistan (e-mail: msikandar59@gmail.com).

DOI: 10.7763/IJMLC.2015.V5.478

problem of *n*=16 and an efficient migrating input network in terms of cost and depth.

Bose and Nelson [5] being one of the pioneers, initiated the research field by presenting a 65 comparator solution for 16 input network. Floyd and Knuth [6] gave a general construction on the problem. Knuth [7], Batcher [4] independently resulted with 63 comparators which was improved by Shapiro [7] with 62 comparisons. Green [8] improved the result by decreasing 2 comparators and is still holding the best result i.e. 60 comparators. No research has ever contributed in decreasing the comparators, however lots of efforts have been done and efficiency has improved, multi-processing concepts improved, and time and overhead reduced. Evolutionary Search took place and holds a good place in the subject. Vooshis [9] came up with merging techniques which need $O(n\log^2 n)$ comparators. However Ajtai *et al.* [10] produces better results O(nlogn) as given by AKS network but it can be said as theoretical result and cannot be achieved practically. Hillis [11] came with 61 comparators with evolutionary search techniques by using first four layers of Green network, thus resulting in the solution for remaining comparators only. Juille [3] improved results of 60 comparators and he also discovered 45 comparators for 13 input problem with one down comparators after many years of deadlock. Evolving Non-Determinism (END) is an incremental algorithm in state spaces search and based on different parameters. Which implies that it will work only in special circumstances?

Graham and Opacher [12] worked on evolutionary search and presented a solution to the symmetric/asymmetric networks with input size of 10. Valsalan and Miikulainen [2] also based on the same principle used by Graham and Opacher [12] and he came up with 16 input , 60 comparator and he also gave results for $16 \le n \le 23$, and claims to improve the result *n*=17, 18,19,20, 21 and 22. Which are the latest and most minimal sets so far?

Knuth [7] improved his previous results by 60, Koza [13], Korenek and Sekanina [14] gave results upto *n*=28, Pursely [15], Parberry [16], [17] also presented the pair wise Sorting. All researchers have matured the field.

TABLE I: RESEARCH CARRIED OUT BY VARIOUS SCHOLARS SINCE 1962 TILL DATE

Sr. No.	**Research by**	**Year**	**Comparators (*n*=16)**	**Algorithm**	**Efficiency**
i.	Bose & Nelson	1962	65	General Construction	$O(n^{1.585})$
ii.	Knuth	1964	63	Zero-One principle	-
iii.	Batcher	1968	63	Odd-Even Merging	$O(\log^2 n)$
iv.	Shapiro	1969	62	-	-
v.	Green	1972	60	-	-
vi.	Van Vooshis	1975	-	-	$O(n\log^2 n)$
vii.	Ajtai	1983	60	-	$O(n\log n)$
viii.	Leighton &Plexton [18]	1990	60	-	-
ix.	Hillis	1991	61	END	-
x.	Jullie	1995	60	Evolutionary	45 for *n*=13
xi.	Graham & Opacher	2006	-	Genetic Algorithm	*N*=10
xii.	Knuth	1998	60	Zero-One principle	-
xiii.	Koza	1999	60	Genetic Algorithm	-
xiv.	Ian Parberry	1992	60	Zero-One principle by Knuth	$O(\log^2 n)$
xv.	Korenk&Sekanina	2005	60	-	-
xvi.	Sung-Soon Choi	2001	60	Genetic Algorithm	Efficient in terms of time
xvii.	Brian Pursley	2008	60	Bitonic Sorters	Cost of $O(n\log^2 n)$
xviii.	Valsalam and Miikkulainen	2013	60	SENSO	Efficient in terms of n>16

To analyze the efficiency of the network, the scholars has made much stress on the size of the network, on minimizing the number of comparators made, network depth, number of parallel machines to be used to carry out the calculation, power issues and optimization of the network.

Ashraf [19] analyzed the number of processors used for Sorting Networks mainly for Batcher-Bitonic, Radix and Quick sort. The previously best known results have been compared with the results of present research and they have improved many times but still needs much more improvement. Valsalan and Miikkulainen [2] have improved the results for inputs of 16, 17, 18, 19, 20, 21 and 22, which give the smallest number of comparators up till now. However, this can further curtail to another minimal set.

III. MAJOR ALGORITHMS USED IN THE PAST

Some Important Algorithms used in the past were Zero-One Principle, Merging, Bitonic Sorters, Genetic Search, Evolutionary Search and END, Batchers Odd-Even, Bitonic Sorter and combination of the different techniques together in different research studies to find the optimal result. The Efficiency of each algorithm is determined in means of Cost and Implementation, while keeping the depth intact for parallel processing.

A. Odd-Even Merging Networks

Batcher [4] presented his work in 1968 on Sorting networks and the applications giving the concept of Odd-even merging networks.

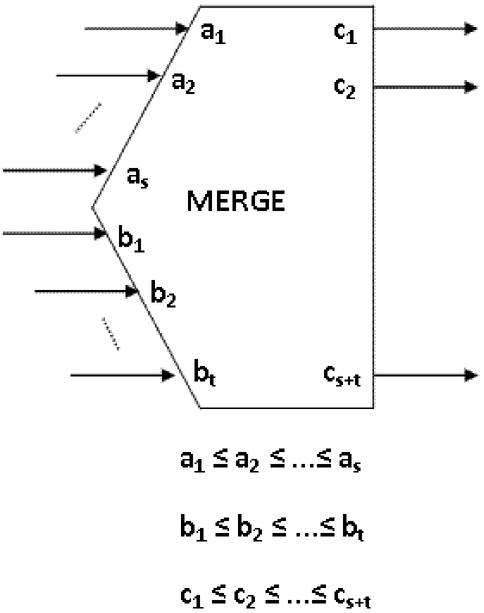
"An odd-even merging network accepts two sequences of length 2^n:

$a_1 \le a_2 \le \cdots \le a_2^n$ and $b_1 \le b_2 \le \cdots \le b_2^n$ and merges them."

B. Zero-One Principal

0-1-Principle, Knuth [1] states that "If a sorting network sorts every sequence of 0's and 1's, then it sorts every arbitrary sequence of values."

i.e. if one mechanism can solve
e.g. 00110010 ->00000111
then it can solve any arbitrary value string.
e.g. 04528671 ->01245678



a_1
a_2
a_s
b_1
b_2
b_t
MERGE
c_1
c_2
c_{s+t}
$a_1 \leq a_2 \leq \ldots \leq a_s$
$b_1 \leq b_2 \leq \ldots \leq b_t$
$c_1 \leq c_2 \leq \ldots \leq c_{s+t}$

Fig. 3. Batcher odd-even merging network.

C. Bitonic Search

Pursely [4] used bitonic sorter to solve the sorting problem. “Abitonic sequenceis a sequence which is first increasing and then decreasing, or can be circularly shifted to become so” and “A bitonic sorter is a comparison network that sort bitonic sequences”.

i.e. 122343321..
(Ascending and then Descending)
or it can be 43321223…
Descending and then Ascending)

D. Genetic Algorithm

Genetic Algorithm [20], [21] used a genetic algorithm with a hybrid steady state, which states the whole network as a chromosome and the genes crossover in two layers used differently for fitness and repair optimization. It was previously studied as a Baldwininian hybrid GA’s [22]-[24].

E. Evolving Non-Determinism (END)

Evolving Non-Determinism (END) as discussed by Hillis [11], can be viewed as a parallel, simulated and evolution search algorithm which minimizes the previous results by 1 for *n*=13. Senso Technique [2] is a two pronged technique that works on symmetric evolutionary Search. .A Gaussian probability distribution has been used by Valasalan to minimize the parent network. According to Valsalam “*It was centered at the middle of its comparator sequence with a standard deviation of one-fourth of its number of comparators. As a result, parent networks were more likely to be truncated near the middle than near the ends. When reconstructing the truncated network, the next comparator to be added to the network was generated either by the estimated model (with probability 0.5) or was selected randomly from the set of all potential comparators (with probability 0.5). Results were insensitive to small changes in these probabilities*.”

F. Migrating Network

Migrating network is the new terminology given for the already being used sorting networks in order to differentiate between other sorting techniques from the comparator based network sorting techniques. The name implies that the inputs are migrating from one to the other horizontal output lines in order to achieve the sorted output. This migration take place several times and increases with the number of total inputs received and very occasionally it happens when the input keeps its original place. To augment the point, consider there will be at least one comparison which needs to be sorted in order to convert the unsorted list into a sorted manner.

IV. Analysis and Discussion

Below is the Table II and Table III giving the number of comparators for each network size ranging from 0-32, where the best results found and verified therefore. The difference between two adjacent numbers being calculated to see where the results differ. It’s a very simple but very effective way to see that where the behavior for the input moves. For input size 17, the comparator size is 71 and for *n*=16, it is 60 which shows that the larger number of comparators is not in the sequence of other results found. However till now, no new result has been proved.

TABLE II: Comparators and Difference between the Comparators for Input Size *N*<=16

Sr. No.	Inputs (*n*)	Comp	Diff
i.	0	0	0
ii.	1	0	0
iii.	2	1	1
iv.	3	3	2
v.	4	5	2
vi.	5	9	4
vii.	6	12	3
viii.	7	16	4
ix.	8	19	3
x.	9	25	6
xi.	10	29	4
xii.	11	35	6
xiii.	12	39	4
xiv.	13	45	6
xv.	14	51	6
xvi.	15	56	5
xvii.	16	60	4

TABLE III: Comparators and Their Inter Differences for Input Size 17<=*N*<=32

Sr. No.	Inputs (*n*)	Comp	Diff
xviii.	17	71	11
xix.	18	78	7
xx.	19	86	8
xxi.	20	92	6
xxii.	21	102	10
xxiii.	22	108	6
xxiv.	23	118	10
xxv.	24	123	5
xxvi.	25	133	10
xxvii.	26	140	7
xxviii.	27	150	10
xxix.	28	156	6
xxx.	29	166	10
xxxi.	30	172	6
xxxii.	31	180	8
xxxiii.	32	185	5

Fig. 4 uses the results of Table II and Table III and shows that there's a continuous increase in comparators as the input size increases, however at only one point where the size increase to 11 which is not according to the previous and even for the next input sizes. Work is being carried out to decrease this comparator to at least 1 point to match the rest of the readings. This deviation is not matching the rest of the data set. And number of comparators, and needs to be reduced.

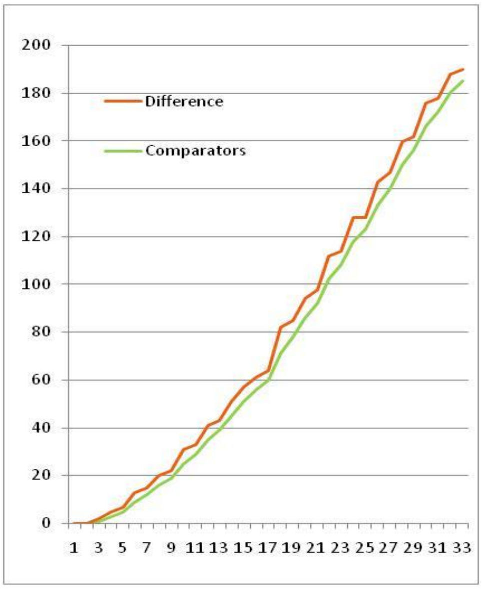


Fig. 4. Continuous increase in Comparators versus the input size difference however all bumps can be seen at the prime number nodes.

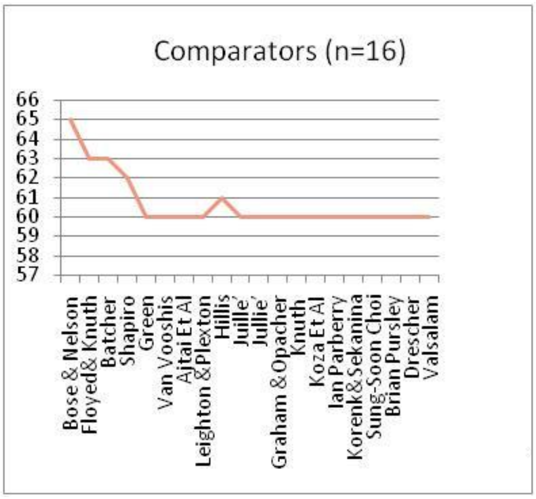


Fig. 5. The number of comparators used by different researchers.

Fig. 5 shows that the research in finding the minimal set of comparators for the input size 16 has not been found since a long time.

V. Results

The study revealed various results driven in the field with the techniques used so far in order to give a view to new researchers who can excel in the field by keeping a framework with them. It mainly focused on the Zero one Principal, END Search, Evolutionary search and Genetic algorithms. However the result shows that most of the work has been done in the Symmetric data networks. Asymmetric search is being neglected which implies that the real world problems need not be symmetric in nature. It is also being observed in the study that only a construction of 0-1 principle, and Odd Even networks served from a long time, however use of Prime number factorization and other series failed or being neglected but can be better optimized. All the algorithms studied have their advantages and some limitations. Need of the day is to contribute towards $n \geq 16$, as the field is already being matured for $n \leq 8$ and defined appropriately for $8 \leq n \leq 16$ for minimized set of comparators. However results can be greatly improved for $n \geq 16$. The results showed that $n=17$ can also be improved at least to one point as the rest of the results implies the same. Secondly the prime number nodes are the main hindrance in evaluating new results.

VI. Future Work

Our future work will mainly address the prime nodes i.e 7,9,11,13,17,21 etc because minimizing their comparators will only produce great results in the field. Evolutionary Search with advancement of genetic algorithms is a promising field to explore. Networks with input $n>16$ are still an open challenge to the new researchers. Complex systems while keeping the implementation cost as a reference to provide a breakthrough to the field with minimal comparators and efficient parallel processing migrating networks. Prime number factorization and genetic algorithm can be great advancement in the field to optimize the migrating network. Fibonacci Series can also help but is not advisable as it works better for the smaller input and not for the larger number of inputs.

References

[1] S. S. Choi and B. R. Moon, "Isomorphism, normalization, and a genetic algorithm for sorting network optimization," in *Proc. Genetic and Evolutionary Computation Conference*, School of Computer Science and Engineering, Seoul National University, Korea, GECCO 2002, 327-334.

[2] V. K. Valsalam and R. Miikkulainen, "Using symmetry and evolutionary search to minimize sorting networks," *JMLR*, 2013, pp. 303-331

[3] H. Juille, "Evolution of non-deterministic incremental algorithms as a new approach for search in state spaces," in *Proc. the 6th International Conference on Genetic Algorithms*, pp. 351-358, San Francisco, CA, USA, 1995, Morgan Kaufmann Publishers Inc.

[4] K. E. Batcher. "Sorting networks and their applications," *AFIPS*, vol. 32, pp. 307-314, April 1968.

[5] R. C. Bose and R. J. Nelson, "A Sorting Problem", *Journal of the ACM*, vol. 9, pp. 282-296, 1962.

[6] R. W. Floyd and D. E. Knuth, "Improved constructions for the Bose-Nelson sorting problem," *Notices of the Amer. Math. Soc*., vol. 14, no. 283, 1967.

[7] D. E. Knuth, *The Art of Computer Programming, Volume III: Sorting and Searching*, Addison-Wesley, 1973.

[8] M. W. Green, "Some improvements in non-adaptive sorting algorithms," in *Proc. the 6th Annual Princeton Conference on Information Sciences and Systems*, 1972, pp. 387-391.

[9] D. C. Van Voorhis, "A generalization of the divide-sort-merge strategy for sorting networks," Technical Report 16, Digital Systems Laboratory, Stanford University, Stanford, California, August 1971.

[10] M. Ajtai, J. Komlos, and E. Szemeredi, "Sorting in *c* log *n* parallel steps," *Combinatorica*, vol. 3, no. 1, pp. 1-19, 1983.

[11] W. D. Hillis, "Co-evolving parasites improve simulated evolution as an optimization procedure," in *Artifical Life II*, J. D. Farmer, C. Langton, S. Rasmussen, and C. Taylor, Eds., Addison, Wesley, Reading, MA, 1991.

[12] L. Graham and F. Oppacher, "Symmetric comparator pairs in the initialization of genetic algorithm populations for sorting networks,"

in *Proc. IEEE Congress on Evolutionary Computation*, 2006, pp. 2845-2850.

[13] J. R. Koza, D. Andre, F. H. Bennett, and M. A. Keane, *Genetic programming III: Darwinian invention and problem solving*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999, ch. 21, pp. 335-348.

[14] J. Korenek and L. Sekanina, "Intrinsic evolution of sorting networks: A novel complete hardware implementation for FPGAs," *Evolvable Systems: From Biology to Hardware*, pp. 46-55, Springer, 2005

[15] B. Pursley. (2008). Sorting Networks. [Online]. Available: http://www.cinlogic.com/Content/Files/CSC204_FinalProject_BrianPursley.pdf

[16] I. Parberry, "A Computer – assisted optimal depth lower bound for nine-input sorting networks," *Mathematical System Theory*, vol. 24, pp. 101-116, 1991.

[17] I. Parberry, "The pairwise sorting network," *Parallel Processing Letters***,** vol. 2, no. 2-3, pp. 205-211, 1992.

[18] F. T. Leighton, "Tight bounds on the complexity of parallel sorting," *IEEE Transactions on Computers*, vol. C-34, no. 4, pp. 344-354, 1985.

[19] A. A. Karim Abu Ain, "Number of processors needed for batcher-bitonic, radix and quick sorting networks methods," *Contemporary Engineering Sciences*, Department of Computer Engineering, Al-Balqa' Applied University, Amman, Jordan, vol. 5, 2012.

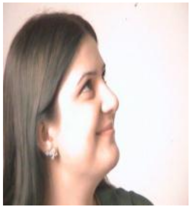
[20] S. S. Choi and B. R. Moon, "A hybrid genetic search for the sorting network problem with evolving parallel layers," *Genetic and Evolutionary Computation Conference*, 2001, pp. 258-265.

[21] S. S. Choi and B. R. Moon, *More Effective Genetic Search for the Sorting Network Problem*, GECCO, 2002, pp. 335-342

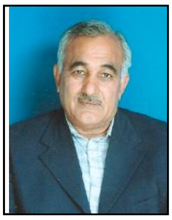
[22] G. G. Simpson, "The Baldiwian effect," *Evolution*, vol. 7, pp. 110-117, 1953.

[23] D. Whitley, V. Gordon, and K. Mathias, "Lamarckian evolution, the Baldwin effect and function optimization," in *Proc. International Conference on Evolutionary Computation*, 1994, pp. 6-15.

[24] G. E. Hinton and S. J. Nowlan, "How learning can guide evolution," *Complex Systems*, vol. 1, no. 3, pp. 495-502, 1987.



Saadia Ureeb is from Preston University, Islamabad. She took her graduation BS degree in computer science and MS degree in computer science from International Islamic University Islamabad in 2004 and 2006 respectively. She started her career by working as an IT manager in a private company, then she served as a lecturer and network administrator in a Computer College for 5 years. Currently She is contributing her sincere and dedicated services to a large public sector organization ZTBL, as an assistant vice president (A.V.P). She is putting her input as a resource person for the staff college of the organization. In addition to her professional studies she is Cisco CCNA (Networking) and CCNA (Security) certified .



Malik Sikandar Hayat Khiyal was born at Khushab, Pakistan, who is currently a professor of Faculty of Computer Science, Preston University, Islamabad. He remained the chairman of Department of Computer Sciences and Software Engineering in Fatima Jinnah Women University Pakistan from 2007 to 2012 and in International Islamic University, Islamabad from 2002 to 2007. He served on the Pakistan Atomic Energy Commission for 25 years (1978-2002) and continuously was involved in different research and development projects of the PAEC. He developed software for underground flow and advanced fluid dynamic techniques. He was also associated with teaching at Computer Training Centre (PAEC) and International Islamic University. His areas of interest are numerical analysis, analysis of algorithms, theory of automata and theory of computation. He has more than hundred and forty research publications to his credit in National and International Journals and Conference proceedings. He has supervised three PhD and more than one hundred and fifty research projects at graduate and postgraduate levels. He is a member of SIAM, ACM, Informing Science Institute, IACSIT. He is an associate editor of IJCTE, IJMO, JACN, LNSE and Coeditor of the journals JATIT and International Journal of Reviews in Computing. He is a reviewer of several Journals, including IJCSIT, JIISIT, IJCEE and CEE of Elsevier.