An Electrical Circuits e-Tutor based on Symbolic and Qualitative Analysis

Jason Debono
Institute for Electronics
Malta College for Science and Technology
Corradino, Malta
jason.debono@mcast.edu.mt

Adrian Muscat
Dept of Communications and Computer Eng.
University of Malta
Msida, Malta
adrian.muscat@um.edu.mt

***Abstract*—Numerical Time Domain electrical circuit simulators are the de facto standard in industry and education. Nevertheless circuits simulators based on symbolic or qualitative techniques have been equally studied. These latter techniques aim at simulating the mental models that an experienced engineer taps when manually designing and analysing a circuit. This paper describes the development of two software tools based on symbolic and qualitative analysis and their use as an electrical circuits eTutor is studied and discussed. As a comparison the paper also investigates the limitations and drawbacks of time domain electrical circuit simulators when used as a pedagogical tool. The field tests are carried out with the engagement of polytechnic teachers and students at the higher national diploma level.**

Keywords-Electrical; Circuits; eTutor; Symbolic; Qualitative, Analysis.

I. INTRODUCTION

Most college, polytechnic and university electrical circuit theory courses include theoretical as well as practical sessions. The practical sessions are important for two reasons; (a) students learn how to link theoretical models to the real-life circuits, and (b) students learn how to carry out the appropriate measurements using the right instrument. These practical skills are indispensable for professional engineers during the installation, testing and maintenance, of electrical and electronics systems. However instrumentation is generally expensive and its use is restricted to labs. In this respect circuit simulators, such as SPICE, augmented with a graphical schematic capture front and back ends are very useful. With such elearning tools students connect virtual components together using virtual wires, choose and add virtual instruments to the circuit, and finally, carry out a computer analyses or. The software outputs the variables chosen or measurements as displayed on the virtual instruments. Such measurements include numerical values, like for example electrical current on a virtual meter, and voltage waveforms on a virtual oscilloscope. This type of eLearning software, widely distributed among colleges and polytechnics helps students in the acquisition of practical skills including the selection of instrumentation. It also speeds up the process and reduces the cost since there is no need for building the circuit in real life. However it does not help the student in understanding how the circuit works or how to design the circuit. On the contrary, it encourages the student not to carry out a manual or mental analysis.

Apart from practical skills, electrical engineering students learn how to analyse and design electrical circuits. Traditionally students have been taught how to analyse electrical circuits using pen and paper through teh application of the relevant theories, including Ohm's Law, Kirchhoff's Current Law and Kirchhoff's Voltage Law. As explained above SPICE simulators were not specifically designed to help students learn how circuits work, consequently SPICE simulators have some serious limitations when used as a pedagogical tool.

The electrical theories taught to students are an essential part of the mental models that the students must develop. Using these theories students can write down symbolic (algebraic) equations that describe how the circuit being analysed behaves. In contrast numerical simulators calculate values iteratively, and this approach limits the understanding and insight that the simulator can impart to its user about how the circuit being analysed functions. In the last few years symbolic simulators have been developed that build the symbolic equations that describe the circuit being analysed and display these equations explicitly. Examples of recently developed symbolic simulators are SAPWIN [1] and SNAP [2].

Additionally, accomplished engineers apply mental models in what-if analysis to understand how a change in a parameter at a point of the circuit affects the other parameters of the circuit. A change in a parameter, like for example the input voltage, is thought of as first influencing the parameters of its neighbouring components and nodes. In turn these varying parameters affect their own neighbours and hence the changes propagate throughout the circuit. This method of analysing a circuit was successfully implemented in a software program in 1977 by Sussman and Stallman, who termed this technique as the 'Propagation of Constraints' [3]. The algorithm implemented by Sussman and Stallman calculated the numerical values of voltages and currents. In fact the mental models used by engineers are usually more simplistic than this because the engineers only consider the direction of change, that is, an increase, a decrease or no change at all in the parameter's value. In other words the quality of the change is considered and not

 ISBN: 978-1-61208-172-4

the quantity of the change. This kind of reasoning has been studied and formally applied to electric circuits and other physical systems like thermodynamic systems in the field of study entitled "Qualitative Physics". In 1984 Johan De Kleer implemented the Qualitative Analysis of electrical circuits in a program he called EQUAL [4]. This program is able to explain how a circuit works using qualitative arguments and even categorize the circuit as being a power-supply, logic-gate, amplifier or multivibrator.

In this paper two programs that target smaller scopes are discussed. The first program accepts input circuits that are made up of an arbitrary number of resistors and only one battery. This program processes serial and parallel connections of resistors. The second program accepts input circuits that are made up of an arbitrary number of resistors, voltage sources and current sources. The software tool then tutors the user on how to select valid spanning trees and the corresponding fundamental cutsets for the input circuit. The symbolic Kirchhoff's Current Law (KCL) equation for each fundamental cutset is then generated by the program, which the user or student can compare to his/her workings. Both tools analyse the topology of the input circuit to accomplish their respective type of analysis. The first program is targeted to MQF level 4 students while the other program is to be used by MQF level 5 students.

The rest of the paper is organised as follows: Section II reviews circuit analysis techniques and section III reviews numerical models in education. The tools developed are described in section IV and the section V discusses results and conclusions.

II. CIRCUIT ANALYSIS PARADIGMS

In this section five types of circuit analysis paradigms are discussed, the *nodal and loop* analysis, *graph theory* in electrical circuits, *qualitative* analysis, *symbolic* analysis and the *propagation of constraints*.

A. Nodal and Loop Analysis

A circuit can be described by using either the mesh or the nodal formulation. The mesh equations are based on Kirchhoff's Voltage Law (KVL), which states that the sum of voltage drops along any closed loop is zero. On the other hand, the nodal equations are based on KCL, which states that the algebraic sum of currents leaving any node is zero. A more general definition of KCL is that in any fundamental cutset that separates the network into two parts, the sum of the currents in the cutset edges is zero. If the number of branches in the network is denoted by the letter *b* and number of ungrounded nodes is denoted by the letter *n*, then to solve a circuit; (a) the number of mesh equations required is equal to $b - n$, and (b) the number of nodal equations required is equal to *n*.

In general nodal analysis yields less equations than mesh (loop) analysis and hence nodal analysis is usually easier to carry out [5].

B. Graph Theory

Graph Theory is used in various ways to aid circuit analysis, for example *Signal Flow Graphs*. This paper focuses on the use of graph theory to analyze the topology of electrical circuits, which is the study of inter-connected objects represented by 'edges' in a graph. The points where the end-points of edges touch together are formally called 'vertices' or 'nodes'.

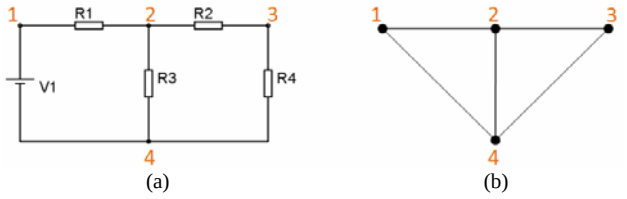

Figure 1: (a) Example Circuit1, and (b) Graph of Example Circuit.

A graph is extracted from the schematic diagram of a circuit by replacing the components with edges. For example the graph shown in fig.1(b) is extracted from the circuit shown in fig.1(a). A graph of an electrical circuit contains more than one spanning tree, and from each spanning tree a set of fundamental loops and fundamental cutsets can be extracted.

1) Spanning Tree

A *spanning tree* of a graph is defined as any set of connecting branches that connects every node to every other node without forming any closed paths or loops [5].

Figure 2: (a) Spanning Tree I, and (b) Spanning Tree II.

Fig.2(a) and fig.2(b) show two different spanning trees for the graph shown in fig.1(b). Once a spanning tree has been defined, the edges making part of the spanning tree are referred to as *branches*. The remaining branches are referred to as *links* or chords.

2) Fundamental Loops

A *fundamental loop* is a loop that contains one (and only one) link in its set of edges [5].

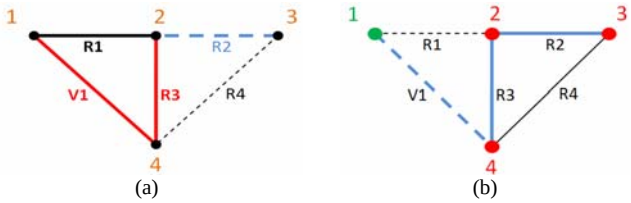

Figure 3. (a) Fundamental Loop for R1, (b) Fundamental Cutset for V1

Fig.3(a) shows the fundamental loop for link R1 when considering the *Spanning Tree* shown in fig.2(b). To construct a loop that includes only the link R1 (which is shown as a thick black line) and no other links, the tree branches shown in red must be used. Therefore the fundamental loop of R1 is made up of the edges: *R1, R3,* and *V1*.

3) Fundamental Cut Sets

A cut set is a minimal set of edges that when cut, divides the graph into two groups of nodes. A *fundamental cutset* is a cutset that contains one (and only one) tree branch in its set of edges [5]. Fig.3(b) shows the fundamental cutset for tree branch V1 when considering the Spanning Tree shown in fig.2(b). By cutting V1 node 1, shown in green becomes isolated from the group of remaining nodes, that is nodes 2, 3 and 4, which are shown in red. Together with branch V1, the link R1 has to be cut to keep the two groups of nodes separated, hence the complete fundamental cutset is: *V1,* and *R1*.

C. Qualitative Electrical Circuits Analysis

De Kleer [4] divides qualitative analysis of electrical circuits into two independent types of analysis, which are; (a) causal analysis, and (b) teleological analysis.

The way that the components are connected together in a circuit gives a specific structure to the circuit. The schematic diagram of a circuit describes this structure. Each component in the circuit causes some effects on the other components that are connected to it, and these in turn affect the components that are connected to them, and so on. The aim of *causal analysis* is to combine the behaviour of the individual components to explain the behaviour of the overall composite system. That said, a composite system is built so that it serves a purpose. The purpose of a circuit is also referred to as the function of the circuit. *Teleological analysis* describes how by knowing the behaviour of a circuit one can deduce its function.

Causal analysis relates *structure* to *behaviour* and *teleological* analysis relates *behaviour* to *function.* These two types of analysis were also investigated by Marc Fosséprez in 1988 [6]. Marc Fosséprez states that it is relatively easy to deduce how a circuit behaves once its function is known, but it is much harder to deduce how a circuit behaves if only the circuit's structure (its schematic diagram) is given. His work focuses on this latter task and he gives definitions about the different structures that circuits can possibly have and mathematical proofs that employ topology and graph theory.

D. Symbolic Simulators

Symbolic simulators are able to generate the transfer function of circuits input to them. The transfer function is a commonly used symbolic expression that describes how a circuit behaves. Using the transfer function the output signal that the circuit generates for a given input signal can be calculated. The advantage of using a symbolic transfer function is that the circuit is analysed symbolically only once to obtain the transfer function and then as many numerical answers as needed can be obtained from the transfer function by substituting the symbols with the numerical values being considered.

Considerable research has been carried out on the symbolic analysis of electrical circuits in the late 1960's and a number of software Symbolic Simulators were developed in the 1980's,[7]. For example De Kleer developed a symbolic simulator called SYN together with Sussman in 1979 [8]. De Kleer states that SYN has several limitations that are were overcome in EQUAL, the Qualitative Analysis Simulator that he developed [4]. These limitations include the lack of the ability to use approximations that drastically simplify the algebra without sacrificing accuracy. Some of these problems have been addressed in modern symbolic simulators [9].

Good examples of modern symbolic simulators that are equipped with a Graphical User Interface (GUI), including a schematic capture front end are SAPWIN [1] and SNAP [2].

E. Propagation of Constraints

The propagation of Constraints has proved to be a powerful algorithm in circuit analysis. Fosséprez recommends its use when searching for a pair of compatible *current* and *voltage* (*i*, *v*) orientations, while analysing circuits qualitatively [6]. In 2006 Peter Robinson et al developed a type of software authoring tool for an 'Intelligent Book' [10] in which this algorithm is used to find the currents, voltages and component values inside different circuits. The values generated by the Propagation of Constraint algorithm are used to verify the values input by the students that make use of the 'Intelligent Book'.

III. NUMERICAL MODELS IN EDUCATION

The main author has carried out a study based on a questionnaire regarding the effectiveness of using SPICE simulators as a pedagogical tool and using handouts which explain step by step the symbolic calculations involved in electric circuit analysis. The questionnaire involved both open ended questions and Likert scale questions. The questionnaire was handed out to the students of the two first year classes of the National Diploma in Electrical and Electronics Engineering (MQF level 4) at MCAST, Malta and a total of thirty one filled in questionnaires were collected. The full report on this study is published in [11]. The report outlines two conclusions that are relevant in this paper; (a) in general although students find the SPICE simulator as motivating very few agree that it helps in understanding how circuits work, and (b) The larger proportion of students acknowledge that it would be much more useful if the simulator explains how the results are obtained. These results confirms what the other researchers that advocate the use of symbolic and qualitative have stated in their papers, which is that software that give explanations, and not just results, aids the students better.

 ISBN: 978-1-61208-172-4

IV. SOFTWARE TOOLS DEVELOPED

Two separate software tools were developed, the first one using MS C++ and the second one using MS C#. The first tool is text based while the second one is provided with a GUI, which makes it more adequate to be used as an eTutor.

A. Input Text Files

Both software tools require the user to input the circuit to be analysed as a text file, but the format is different in the two cases. The first tool requires the circuit to be specified in a matrix in which the rows represent nodes and columns represent components. The first row of this matrix is reserved for the components' values, that is the voltage of the battery and the resistance of each resistor. In the cells of the other rows a '1' means that terminal one of the corresponding component is connected to the node corresponding to the cell's row, '2' means that terminal two of the corresponding component is connected to the node corresponding to the cell's row and '0' means that the corresponding component is not connected to the particular node considered. The format of the input text file for the second program is more compact since in it a text line is dedicated to each component and the numbers of the two nodes to which the component is connected are stated in the corresponding line. This does away with the '0s' that were used for the first program. The other information included in each line of this text file is the X and Y coordinates of where the component is to be drawn in the GUI, the name of the component and its value.

B. Series and Parallel Reductions Program

The first software tool accepts input circuits that are made up of an arbitrary number of resistors and only one battery. This program then analyses the connections of all the resistors and identifies resistors that are connected in parallel. Each group of resistors connected in parallel is replaced by one equivalent resistor. The program then identifies serially connected resistors and replaces each group by one equivalent resistor. This process is depicted in fig.4. At each step the program outputs a matrix in text format which specifies the connections in the resultant simplified circuit.

If the resultant circuit contains other groups of resistors that are connected in parallel or in series further reductions are done. This process is repeated until no further reductions are possible.

C. Graphical Circuits Analysis Program

The aim of the program is to eTutor students that are learning how to identify a fundamental tree and the corresponding fundamental cutsets in a given circuit and how to generate the KCL current equations for each fundamental cutset. This topic is covered in a unit called 'Further Electrical Principles' that higher national diploma (MQF level 5) students follow in the second year of their course at MCAST.

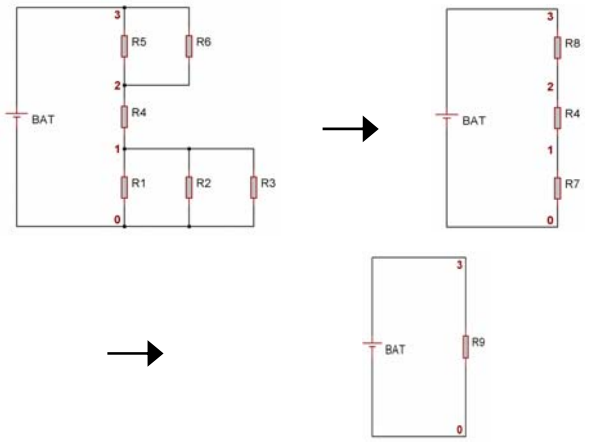

Figure 4. Example of the parallel and series resistors reduction processes carried out by the first program.

Once a circuit is specified correctly in the input text file it can be loaded in the program. Fig 5. shows an example of a loaded circuit. The user is asked to chose a spanning tree, by clicking on the components in the circuit. Once the user selects a group of components that s/he think makes up a valid spanning tree, s/he must press the 'Check Spanning Tree' button so that the program verifies if the selected group of components makes up a valid spanning tree. If it does not the program informs the user and gives relevant feedback to the user of why the selection does not make up a valid spanning tree. The program informs the user whether s/he selected the right amount of components and whether s/he captured all the nodes in the circuit with the group of components selected. The program also informs the user if there are loops present in the selection made.

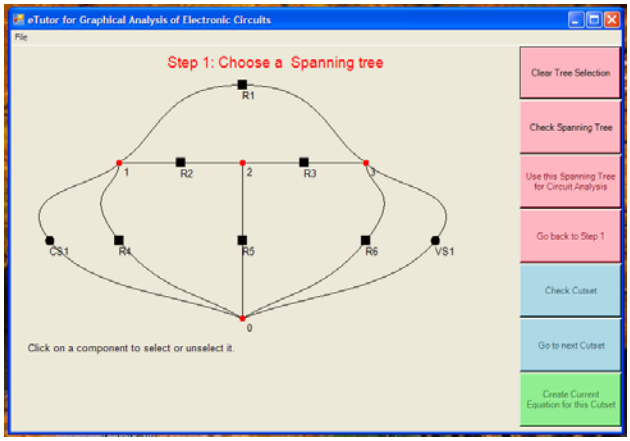

Figure 5. Example of a loaded circuit in the program's GUI

On the other hand, if the selection makes up a valid tree the program informs the user and allows the user to select this spanning tree to continue with the circuit analysis. To do this the user has to press the 'Use this Spanning Tree for Circuit Analysis' button. Once this button is pressed the program goes into Step 2, in which the user has to select the

 ISBN: 978-1-61208-172-4

correct fundamental cutset for each of the tree branches inside the selected spanning tree. The tree branch for which the user has to select the links that make up the fundamental cutset is highlighted in red, as shown in fig.6.

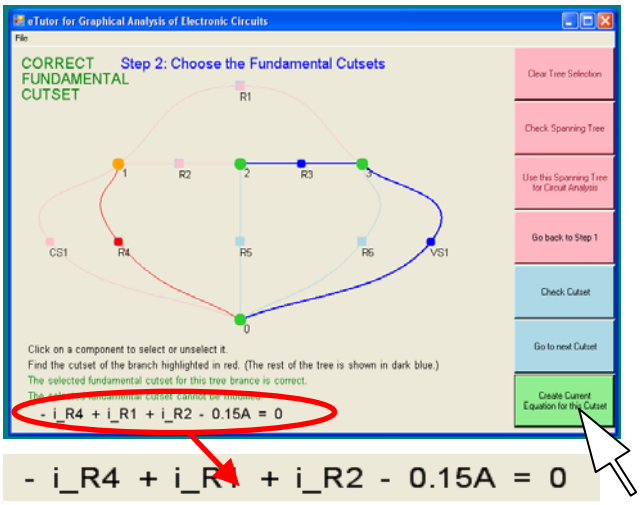

Figure 6. Example of a fundamental cutset KCL equation generated when the correct fundamental cutset is selected and the appropriate button is pressed

The fundamental cutset must separate one of the group of nodes from the remaining group of nodes.To help the user the program highlights all the nodes in one of these groups in orange and the nodes in the other group in green. After that the user selects the components that s/he thinks make up the fundamental cutset, s/he must press the 'Check Fundamental Cutset' button. Once this button is pressed the program checks if the selected components make up a valid fundamental cutset. If this is not the case the program gives relevant feedback to the user. The program states whether one or more components that should be included in the selection are not selected and it also states if one or more components that should not be included in the selection are in fact selected. In the case when the selected components make up a valid fundamental cutset, then the user is informed accordingly and is allowed to press the button labelled "Create Current Equation for the Cutset". When this button is pressed the KCL equation for the fundamental cutset is generated by the program and displayed at the bottom of the screen as shown in fig.6. The user can then press the 'Go to next Cutset' button to find the fundamental Cutset of the next branch in the spanning tree. This process has to be repeated until the fundamental cutsets of all the branches in the spanning tree are found.

At this point it is desirable that the program tutors the user on how to find the fundamental loop for each link present in the graph, but this feature has not been implemented yet. The author plans to have this feature functional in the future so that it can be used by the higher national diploma students.

D. Algorithms in the Graphical Circuits Analysis Program

In the computer program developed, the nodes are implemented in a list. The branches or components at a given node are defined in another list. The algorithms then operate on these lists. From graph theory it is known that a valid spanning tree must be made up of *n-1* edges, where *n* is the number of nodes. Hence the first check made to verify the input tentative spanning tree is to count the number of selected components and check if it equal to *n-1*. If this is not the case it means that the selected components do not make up a valid spanning tree.

The next step to carry out is to check that the selected components capture all the nodes inside the circuit (graph). The algorithm just has to go through all the selected components and mark the two nodes to which each component is connected as *captured*. After that the algorithm has to go through the nodes and check that none of them is *non-captured*. If one or more nodes are *non-captured* then the selected components do not make up a valid spanning tree. There exist cases in which the two checks explained above are satisfied but the selected components still do not make up a valid spanning tree. In this case the selected branches will not be continuously connected and at least one loop will be present in the selection. To check for such cases the spanning tree algorithm starts off with one of the selected tree branches. It checks to which nodes this branch is connected and proceeds to discover which other branches one of these nodes is connected to. If there are more than one branch connected to this node the algorithm starts considering the first branch and it takes note of which branch this is so that once it finishes checking it and returns to the last node considered, it continues looking for the correct branch. This process is repeated for each node. When at least one branch is found connected to a node the algorithm jumps to the other node to which this branch is connected and hence travels further away from the first node that it considered at the start. Naturally the larger the selected tentative spanning tree is, the more searching the algorithm has to do. But in the case of invalid spanning tree selections there are two possible ways in which the algorithm completes. One way is that the algorithm steps forward (not backwards) into a node that it already checked, and hence a loop is discovered. The other way in which the algorithm can complete in the case of an invalid spanning tree selection is that it finds out that it exhausted all the branches and nodes that are connected to the first branch considered, but it did not find all the nodes present inside the graph. In this case it means that the algorithm has found one continuous length of connected branches, which is not connected to the remaining branches of the selected tentative spanning trees. Since spanning trees should not contain any discontinuities in their branches' connection, this means that the selected components do not make up a valid spanning tree.

Another algorithm used in the graphical analysis program is the one that highlights in different colours the

 ISBN: 978-1-61208-172-4

two groups of nodes that are to be separated by a fundamental cutset. The searching that this algorithm does is very similar to that done by the algorithm that verifies spanning trees. However in this case, the fundamental cutsets algorithm does not check for loops because it is used after that a valid spanning tree is already selected, so it is already guaranteed that no loops are present. The important feature that this algorithm possesses, similarly to the previous algorithm, is that it always remembers which branch it checked last when jumping from one node to another, so that when it returns back to the node from where it jumped, it continues checking from the correct branch.

V. CONCLUSIONS

An important conclusion from the questionnaire delivered to the national diploma (MQF level 4) students is that they are keen to experience software tools that help them understand how to analyse electrical circuits, by explaining the results that these programs generate. This type of software simulates the full-time availability of a tutor.

Two software tools that give explanations of the analysis carried out computationally on circuits were developed. Both these programs contain elements of Symbolic and Qualitative analysis.

One of these programs is aimed at first year national diploma (MQF level 4) students. It identifies resistors that are connected in parallel and in series and replaces them by equivalent resistors. This program was tested and verified to function correctly, but since it was developed as a text based program it is not easy for the students to use it. Hence an improvement that is needed for this program to become useful is the provision of a GUI.

The other tool is aimed at second year higher national diploma (MQF level 5) students. Its aim is to tutor these students on how to find correct spanning trees and fundamental cutsets in graphs of electrical circuits. This program was tested and verified to function correctly. It interacts with its users through a GUI. It lets the user input the circuit of interest and try to select a valid spanning tree. When a valid spanning tree is selected the program lets the user work out all the valid fundamental cutsets corresponding to this spanning tree and then generates the corresponding KCL equations. Whenever the user does an incorrect choice during the selection process, the tool explains why the choice is incorrect, and hence acts like a Tutor.

There are many possible improvements that can be done to this tool, the most important of which is the inclusion of fundamental loops selections. The feedback that this program gives to its user can also be more informative, or even better, be in increasing steps of information, according to how much help the user desires to get. In any case, even at the current stage of development this program can aid significantly the students that are learning this topic. Besides being used by the author, this program was demonstrated to two lecturers that teach this topic and both confirmed that this program will help them deliver the concerned topic more efficiently, leading to higher success rates among students.

The tool was first tested by fifteen students that undertook courses that included the topic under consideration in the previous academic year and all these students stated that this tool would have been of great help to them. The program was then tested with a class of 18 novel HND students during the academic year (2010-2011). These students were able to choose their own personal set of branches that make up valid spanning trees and fundamental cutsets in class. This reduced the amount of time that the students needed to learn and understand these two concepts, as well as the success rate among students.

REFERENCES

[1] A. Luchetta, S. Manetti and A. Reatti, “SAPWIN - A Symbolic Simulator as a Support in Electrical Engineering Education”, IEEE Transactions on Education, Vol. 44, pp. 9, May 2001.

[2] D. BIOLEK, “SNAP – program with symbolic core for educational purposes”, Proceedings of 4th World Multi-Conference on: Circuits, Systems, Communications and Computers, ISBN 960-8052-19-X, pp. 1711-1714, July 2000.

[3] G. Sussman and R. Stallman “Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis”, Artificial Intelligence, Vol. 9, pp. 135-196, October 1977.

[4] J. de Kleer, “How circuits work”, Artificial Intelligence, Special volume on qualitative reasoning about physical systems, Vol. 24, pp. 205-280, December 1984.

[5] J. W. Nilsson and S. A. Riedel, Electric Circuits 5th Edition, Addison Wesley, 1996.

[6] M. Fosséprez, Qualitative Analysis of Non-linear, Non-reciprocal Circuits, John Wiley & sons, 1992.

[7] G. Gielen, P. Wambacq and W.M. Sansen, “Symbolic Analysis Methods and Applications for Analog Circuits: A Tutorial Overview”, Proceedings of the IEEE, vol. 82, pp.287-304, 1994.

[8] J. de Kleer and G. Sussman, “Propagation of constraints applied to circuit synthesis”, International Journal of Circuit Theory and Applications, Vol. 8, pp. 127–144, April 1980.

[9] H. Floberg, Symbolic Analysis in Analog Integrated Circuit Design, Kluwer Academic Publishers, 1997.

[10] K. Rehman, W. Billingsley, and P. Robinson, “Writing Questions for an Intelligent Book Using External AI”, Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies, ISBN 0-7695-2632-2, pp. 1089 - 1091, 2006.

[11] J. Debono, “Effectiveness of using Circuit Analysis Software in Vocational Electronics Engineering Courses”, Malta College of Arts, Science and Technology (MCAST) library, September 2010. Also available at: https://sites.google.com/site/jasondebonoeportfolio/o

 ISBN: 978-1-61208-172-4