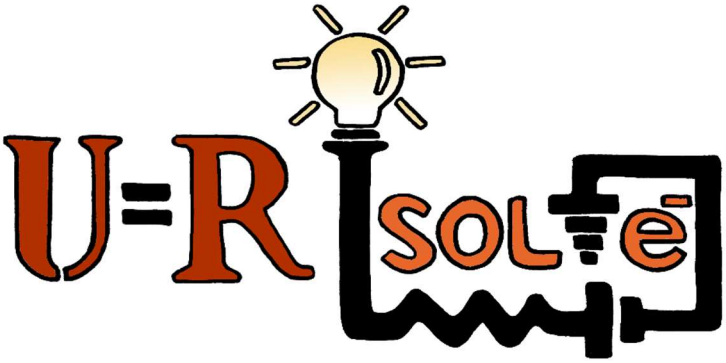
APLICAÇÃO PARA GERAÇÃO DA METODOLOGIA/EQUAÇÕES DO MÉTODO DAS CORRENTES NOS RAMOS A PARTIR DO QUCS



Joshua Williams Soares

isep Instituto Superior de Engenharia do Porto

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

2019

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Unidade Curricular de Projeto/Estágio, do 3º ano, da Licenciatura em Engenharia Eletrotécnica e de Computadores

Candidato: Joshua Williams Soares, Nº 1141227, 1141227@isep.ipp.pt

Orientador: Mário Jorge de Andrade Ferreira Alves, mjf@isep.ipp.pt

Coorientador: Francisco José Dias Pereira, fdp@isep.ipp.pt

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

24 de outubro de 2019

Agradecimentos

Este projeto é o resultado de meses de trabalho. Mas este trabalho não teria sido possível sem a ajuda de várias pessoas que ao longo do mesmo, de uma forma ou de outra, contribuíram para que fosse possível. Por isso quero agradecer aqui em especial:

Aos orientadores do projeto, Professor Mário Alves e Professor Francisco Pereira pela dedicação, orientação e acompanhamento.

À minha família pela motivação e pela forma como sempre me incentivaram a superar todos os desafios que vão surgindo ao longo da minha vida.

Aos colegas Lino Sousa e Miguel Duarte pela ajuda e disponibilidade.

Resumo

Com este projeto pretendeu-se desenvolver uma aplicação que permitisse a análise de circuitos elétricos. Ao contrário dos simuladores existentes, em que os resultados devolvidos aos utilizadores se traduzem em valores mensurados, através desta aplicação obtêm-se as equações matemáticas que subjazem à análise de circuitos elétricos em Corrente Contínua e Corrente Alternada com recurso ao Método das Correntes nos Ramos (MCR).

Dessa forma e tendo acesso às equações matemáticas, é possível aos utilizadores uma maior aprendizagem que pode ser usada com facilidade na resolução de circuitos elétricos em Corrente Contínua e Corrente Alternada com recurso ao Método das Correntes nos Ramos (MCR).

Recorrendo ao simulador *Quite Universal Circuit Simualtor* (QUCS) obtém-se um ficheiro de texto – a *netlist* – que de seguida é carregado para a aplicação desenvolvida. A aplicação lê o ficheiro, resolve o circuito e apresenta os resultados sob a forma de equações matemáticas obtidas através da análise do MCR.

Tendo como público alvo estudantes de Engenharia Eletrotécnica (EE), pretendeu-se igualmente que esta aplicação fosse abrangente no número de utilizadores, daí que se tenha optado por uma arquitetura simples e acessível: desenvolvimento em *HyperText Markup Language* (HTML) e a implementação da análise computacional em JavaScript (JS).

Palavras-Chave

Análise de Malhas, Leis de Kirchhoff, Equações das Malhas, Simulador de Circuitos, Equações dos Nós, QUCS, JavaScript, HTML, Engenharia Eletrotécnica, Equações dos Nós.

Abstract

This project aimed to develop an application that allowed the analysis of electrical circuits. Unlike existing simulators, where the results returned to users are measured values, through this application we obtain the mathematical equations that underlie the analysis of electrical circuits in Direct Current and Alternating Current using the currents branches method (MCR).

Thus, having access to mathematical equations, it is possible for users to achieve more knowledgement that can be easily used in the resolution of electrical circuits in Direct Current and Alternating Current using the currents branches method (MCR).

Using the Quite Universal Circuit Simulator (QUCS), a text file is generated - the *netlist* - which is then uploaded to the application developed here. The application reads the file, solves the circuit, and presents the results in the form of mathematical equations obtained through MCR analysis.

Targeting students of Electrotechnical Engineering (EE), it was also intended that this application had a comprehensive number of users, so it was decided to use a simple and accessible architecture: development in HyperText Markup Language (HTML) and the implementation of computational analysis in JavaScript (JS).

Keywords

Mesh Analysis, Kirchhoff Laws, Mesh Analysis, Circuit Simulator, QUCS, Node Equations, JavaScript, HTML, Electrical Engeneering

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

HTML	-	HyperText Markup Language
JS	-	JavaScript
PCB	-	Printed Circuit Board
CSS	-	Cascading Style Sheets
LKN	-	Lei de Kirchhoff dos Nós
MCR	-	Método das Correntes nos Ramos
EE	-	Engenharia Eletrotécnica
MNA	-	Modified Nodal Analysis
MTN	-	Método das Tensões nos Nós
OSS	-	Open Source Software
QUCS	-	Quite Universal Circuit Simulator

1. INTRODUÇÃO

Neste capítulo falar-se-á do contexto em que o presente projeto surge e dos objectivos do mesmo, expondo-se as contribuições e finalidades.

Far-se-á uma apresentação do presente relatório, demonstrando-se a cronologia dos trabalhos, nomeadamente quanto à sua estrutura interna permitindo aos leitores conhecer os capítulos em que o mesmo se divide e os temas abordados em cada capítulo.

1.1. CONTEXTUALIZAÇÃO

A temática dos circuitos elétricos apresenta-se com um grau de dificuldade elevado para os alunos que dão os primeiros passos na EE. Trata-se de uma área chave, que serve de base a demais aprendizagens exigidas para que num futuro os atuais estudantes tenham sucesso. É pois primordial que os conceitos dessa temática sejam interiorizados e percebidos de forma correta, para que os mesmos possam ser aplicados sem dificuldades e sem erros.

No processo de ensino aprendizagem é sempre essencial tomarmos como ponto de partida o do próprio estudante, visto ser ele que se depara com as reais dificuldades. Estas, no âmbito dos circuitos elétricos, passam por identificar corretamente os Ramos, os Nós, as Malhas, e tornam-se um problema ainda maior quando é necessário transpor as variáveis para uma equação relativa a um Nó ou uma Malha completa.

Existem ferramentas que permitem diminuir essas dificuldades; falamos dos simuladores existentes (que serão alvo de análise no capítulo seguinte) e que efetivamente são úteis para testar o circuito em si e para devolver os resultados. No entanto, estes não fornecem aos estudantes o "caminho" do circuito; dito por outras palavras os simuladores oferecem as soluções mas não mostram os passos necessários para que se alcancem as soluções. Uma aprendizagem que se foque nos passos a dar, nas equações a resolver, passo a passo, para se obter um resultado final, terá certamente mais sucesso que aquela que apresenta as soluções sem mostrar e demonstrar o caminho.

A aplicação desenvolvida pretende colmatar esse "vazio" – o caminho entre o circuito e o resultado. Mostrando esse caminho, isto é as equações matemáticas que estão na base dos circuitos elétricos, a aplicação assume um carácter didático, colocando o aluno numa posição ativa didática no processo de ensino/aprendizagem.

1.2. OBJETIVOS

Tendo por base o contexto em cima referido, pretende-se com esta aplicação alcançar três objetivos:

- envolver os alunos na aprendizagem de circuitos elétricos, compreendendo os passos necessários para que aqueles funcionem corretamente e conforme pretendido, possam obter melhores resultados académicos e, posteriormente, profissionais;
- complementar as ferramentas de análise de circuitos elétricos existentes, visto que na sua maioria essas ferramentas não demonstram os passos necessários/equações matemáticas subjacentes ao bom funcionamento de um circuito elétrico;

- e incluir na aplicação U=RIsolve, já elaborada por outros estudantes [1][2], o módulo de geração de equações do Método das Correntes nos Ramos, melhorando o algoritmo já criado de aquisição de dados e aprimorando o *layout* da plataforma e da página de resultados.

É de frisar que a aplicação U=RISOLVE não pretende substituir ou alterar o método de ensino, mas sim complementá-lo. Dito de outra forma, esta aplicação aproveita-se de informação inerente a cada circuito, obtida a partir do *QUCS*, que é um simulador frequentemente utilizado no ensino da EE e já melhorado por outros estudantes [3][4], e recorrentemente alvo de estudo e desenvolvimento devido à sua característica de *OpenSource Software* (OSS), introduzindo uma componente pedagógica, externa a este *software*, de forma a reforçar a aprendizagem.

Este projeto, como já foi dito anteriormente, visa o estudo do MCR e a sua respetiva aplicação na análise de circuito elétricos simulados no QUCS. Dele, pretende-se que resultem: o arbítrio das Correntes nos Ramos, as equações do Nós e as equações das Malhas, tanto na análise de circuitos em Corrente Contínua bem como em circuitos em Corrente Alternada.

O grande desafio deste projeto reside no desenvolvimento de um algoritmo que seja capaz de encontrar, por si só, o número de Malhas suficientes, de validá-las e selecioná-las automaticamente para uma posterior utilização na resolução do MCR.

1.3. CALENDARIZAÇÃO

Este projeto, desenvolvido ao longo de vinte e sete semanas. De referir que se podem autonomizar oito etapas, as quais possuíam objetivos próprios, com graus de dificuldade diferentes. A parte inicial envolveu o estudo do MCR, do método de geração de *netlist* por parte do QUCS e a revisão de conhecimentos referentes às linguagens necessárias para o projeto: JS, HTLM e CSS. Posteriormente procedeu-se ao desenvolvimento da aplicação, à sua melhoria e aos testes à mesma. Por último elaborou-se o presente relatório.

Tabela 1 Calendarização do projeto

ID	Nome das Etapas	Início	Fim	Duração	Abril 2019					Maio 2019				Junho 2019				Julho 2019					Agosto 2019				Setembro 2019					Outubro 2019			
					1/4	8/4	15/4	22/4	29/4	6/5	13/5	20/5	27/5	3/6	10/6	17/6	24/6	1/7	8/7	15/7	22/7	29/7	5/8	12/8	19/8	26/8	2/9	9/9	16/9	23/9	30/9	7/10	14/10	21/10	28/10
1	Estudo do MCR	01/04/2019	14/04/2019	2w																														
2	Estudo do método de geração netlist pelo QUCS	15/04/2019	05/05/2019	3w																														
3	Revisões das linguagens HTML, CSS e JS	06/05/2019	26/05/2019	3w																														
4	Desenvolvimento da aplicação	06/05/2019	01/09/2019	17w																														
5	Pesquisa de soluções para problema de geração de malhas	2705/19	14/07/2019	7w																														
6	Melhorias no funcionamento e apresentação de resultados	26/08/2019	15/09/2019	3w																														
7	Fase de testes	02/09/2019	22/09/2019	7w																														
8	Elaboração do relatório final	09/09/2019	06/10/2019	7w																														

1.4. ORGANIZAÇÃO DO RELATÓRIO

O presente relatório encontra-se divido em seis capítulos. No primeiro capítulo expõe-se os motivos para a escolha deste projeto, procurando-se evidenciar as contribuições do mesmo do ponto de vista científico, tecnológico e didático. Igualmente neste capítulo são indicadas, de forma cronológica, as etapas e calendarização deste projeto, através do recurso a um diagrama de Gantt.

O segundo capítulo debruça-se sobre os simuladores de circuitos elétricos existentes na atualidade, caracterizando-os de forma resumida. No final deste capítulo encontra-se uma tabela comparativa dos simuladores referidos e sublinha-se a escolha do QUCS como a melhor ferramenta para uso em contexto académico.

O terceiro capítulo refere-se às metodologias de análise de circuitos elétricos através do Método de Correntes nos Ramos, terminando com o recurso a exemplos de análises de circuitos através desse método.

O quarto capítulo é dedicado exclusivamente ao funcionamento da aplicação, descrevendo de forma exaustiva todos processos que foram importantes para alcançar o produto final: o módulo de geração das equações/metodologia para o método das Correntes nos Ramos a partir do QUCS, na aplicação U=RIsolve.

No quinto capítulo são demonstrados os resultados da aplicação, expondo circuitos exemplo em Corrente Contínua e Corrente Alternada e, para o primeiro caso, a respetiva validação dos resultados.

Finalmente, no sexto capítulo é feita uma conclusão, onde se abordam os objetivos alcançados e as dificuldades sentidas.

2. SIMULADORES DE CIRCUITOS ELÉTRICOS

Atendendo ao trabalho ora realizado, os simuladores de circuitos elétricos que passaremos a descrever estão vocacionados para o trabalho académico e apoio aos estudantes no seu percurso de autoaprendizagem, valorizando dessa forma a autonomia dos alunos. Procuramos descrever as suas características, as vantagens e desvantagens de cada um deles.

2.1. SIMULADORES DE CIRCUITOS ELÉTRICOS EXISTENTES

CircuitLab

O *CircuitLab* [5] é um simulador em *browser,* que possui quer uma versão gratuita (para fins não comerciais), quer versões pagas. Este simulador é implementado em JavaScript, sendo por isso desnecessário a instalação de um *plug-in*. A versão gratuita disponibiliza ferramentas e componentes básicos de simulação úteis para quem inicia o percurso no estudo da engenharia eletrónica. Destaca-se a interface de utilização: simples e fácil de usar. As versões pagas deste software variam entre 20 e 80 euros e, naturalmente, oferecem mais funcionalidades, documentação e tutoriais.

Uma das grandes vantagens deste simulador é a existência de um fórum para os utilizadores, sendo possível a estes expor dúvidas ou procurar soluções para eventuais problemas encontrados.

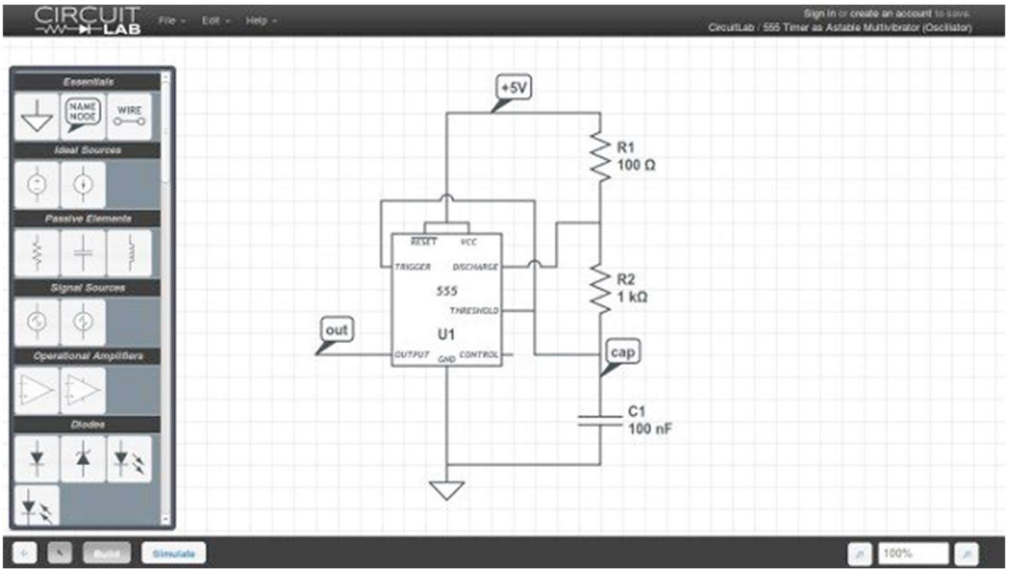


Figura 1 Exemplo da aplicação *CircuitLab* [5]

EasyEDA

Este simulador, *EasyEDA* [6]**,** apresenta-se como simples e interativo. Pode ser obtido para sistemas Linux, macOS e Windows ou usado em plataforma *web* em *browser*. A licença é gratuita, possui atualizações semanais e várias bibliotecas de componentes e módulos de circuitos *standard*.

Este software tem como vantagem a conversão em formato PCB do esquemático que resulta da simulação. É possível importar ficheiros esquemáticos de circuitos e possui vários vídeos tutoriais que permitem a autoaprendizagem por parte dos alunos que recorrem ao mesmo.

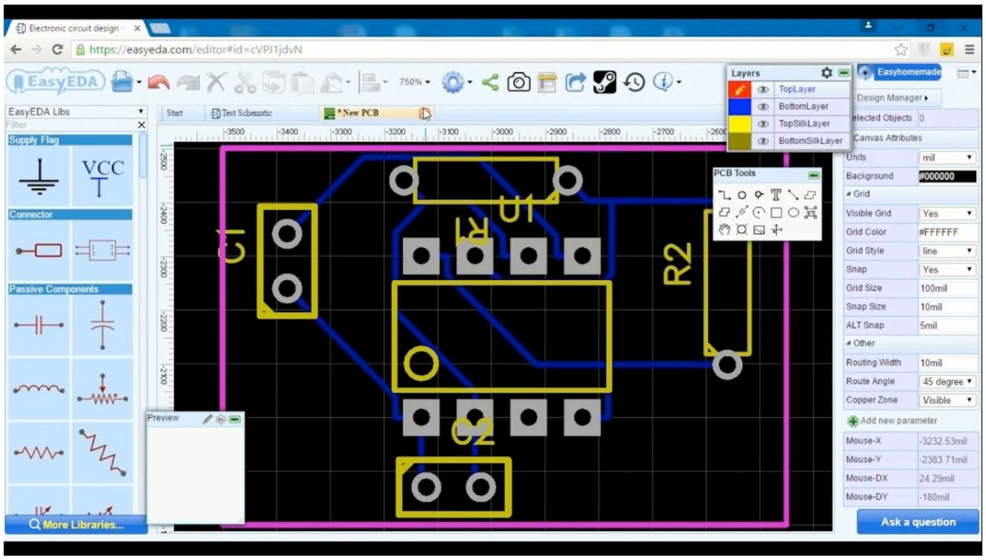


Figura 2 Exemplo da aplicação *EasyEDA* [6]

EveryCircuit

O *EveryCircuit* [7] foi inicialmente desenvolvido para iOS e *Android* (onde permanece gratuito), sendo que atualmente existe uma versão em browser paga, que possui bibliotecas mais completas. A interface gráfica é apelativa, permitindo uma visualização dinâmica e em tempo real de gráficos de medição e fluxos de Corrente no circuito simulado.

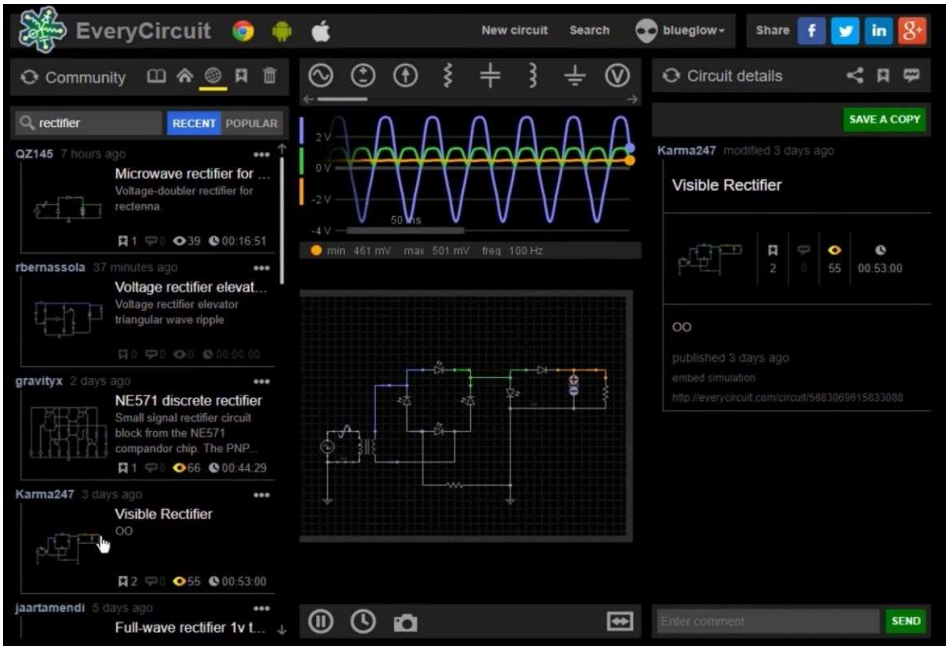


Figura 3 Exemplo da aplicação *EveryCircuit* [7]

KTechLab

O *KTechLab* [8] é uma aplicação *Open Source* que foi desenvolvido apenas para Linux. Este simulador permite a simulação de circuitos elétricos e também a simular projetos de eletrónica digital com microcontroladores, desenvolvendo o programa e o circuito. A sua interface é intuitiva e simples permitindo o seu uso por novos utilizadores.

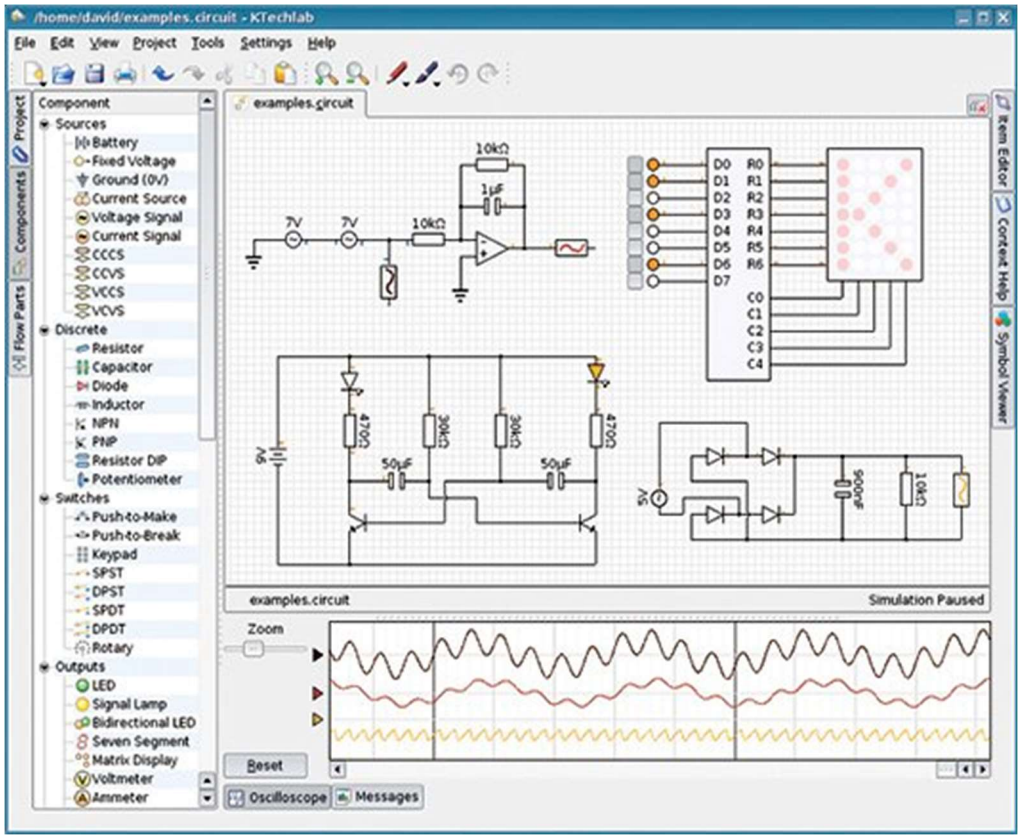


Figura 4 Exemplo da aplicação *KTechLab* [8]

LTspice

O *LTspice* [9], é uma ferramenta *freeware* disponível para macOS e Windows, robusta mas leve, utilizando poucos recursos da máquina onde se encontra instalado e tornando-se, por isso rápido. Estas características fazem com que este *software* seja, possivelmente, *o* mais usado pela comunidade académica, embora a interface não seja apelativa. Possui atualizações mensais.

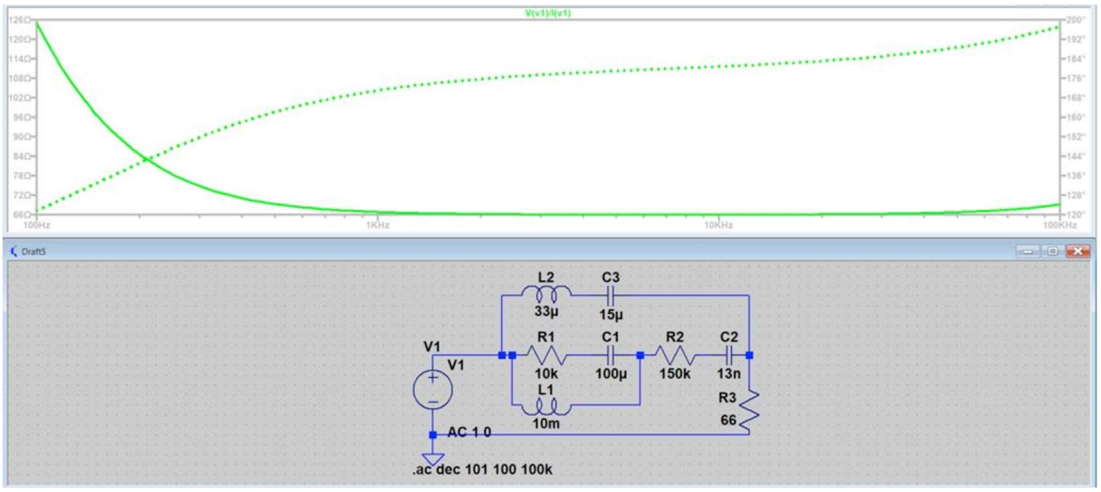


Figura 5 Exemplo da aplicação *LTSpice* [9]

Oregano

O *Oregano* [10] é um OSS (*Open Source Software*) e encontra-se apenas disponível para sistemas *Linux*. Possui uma biblioteca de componentes bastante completa, que é atualizada regularmente. A interface é básica e simples, o que pode auxiliar os utilizadores menos experientes. No entanto quando se tratam de simular circuitos complexos, esta aplicação não é a mais indicada porque a sua simplicidade não permite analisar aquela complexidade.

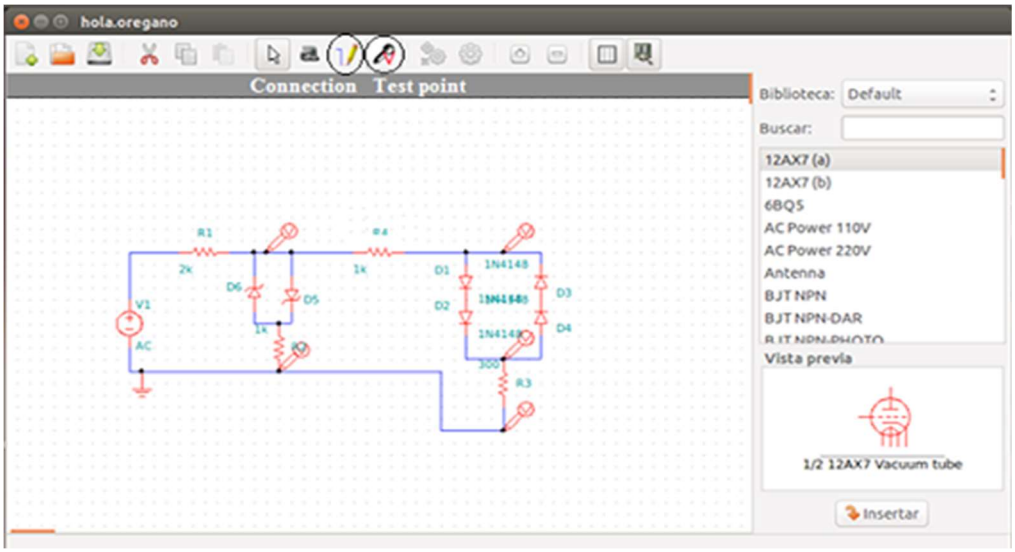


Figura 6 Exemplo da aplicação *Oregano* [10]

Falstad Circuit Simulator

Este simulador, *open-source,* é implementado em browser e apresenta-se como simples e intuitivo. O *Falstad Circuit Simulator* [11] permite visualizar o fluxo de Corrente e bem assim a forma como os componentes o alteram em tempo real. Ou seja, possibilita-se ao aluno controlar o circuito em tempo real. A interface é apelativa e contém vários exemplos de circuitos.

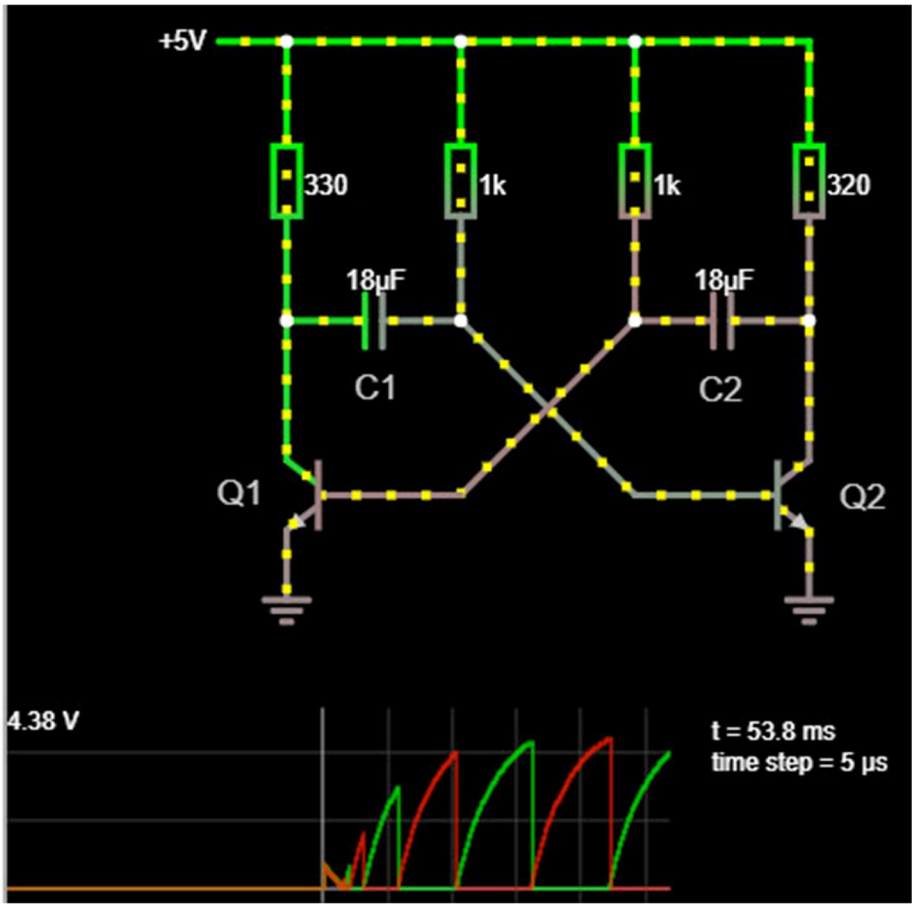


Figura 7 Exemplo da aplicação Falstad Circuit Simulator [11]

PSIM

O PSIM [12] é um programa apenas existente para *Microsoft Windows*. Apesar de ser pago, existe uma versão experimental de 30 dias, na qual poderão ser testadas todas as funcionalidades acessíveis. A limitação desta versão experimental diz respeito ao tamanho do circuito, não sendo possível simular um circuito com mais de 34 componentes. A é relativamente simples e bastante similar à de outros simuladores.

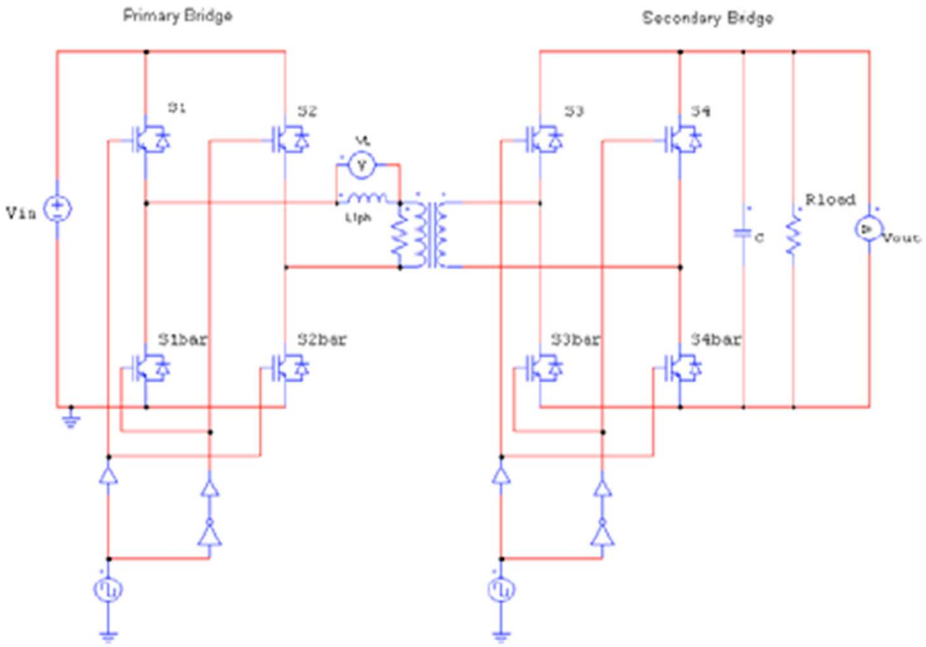


Figura 8 Exemplo da aplicação PSIM [12]

PSpice

O *PSpice* [13], da *Cadence*, é possivelmente o *software* de simulação de circuitos mais completo do mercado; possui uma vasta biblioteca de componentes e modelos de circuitos. A versão de compra única custa 6000€, e engloba 6 aplicações, a *OrCAD Capture*, *OrCAD Capture CIS*, *PSpice A/D*, *PSpice Advanced Analysis*, *OrCAD PCB* e *SPECCTRA*. Este simulador está mais vocacionado para um uso profissional, o que é demonstrado quer pelo preço quer pela interface que se mostra pouco apelativa

Existe a possibilidade de *download* da versão *trial* (30 dias) e da versão *lite*, que limita o uso de componentes mais complexos.

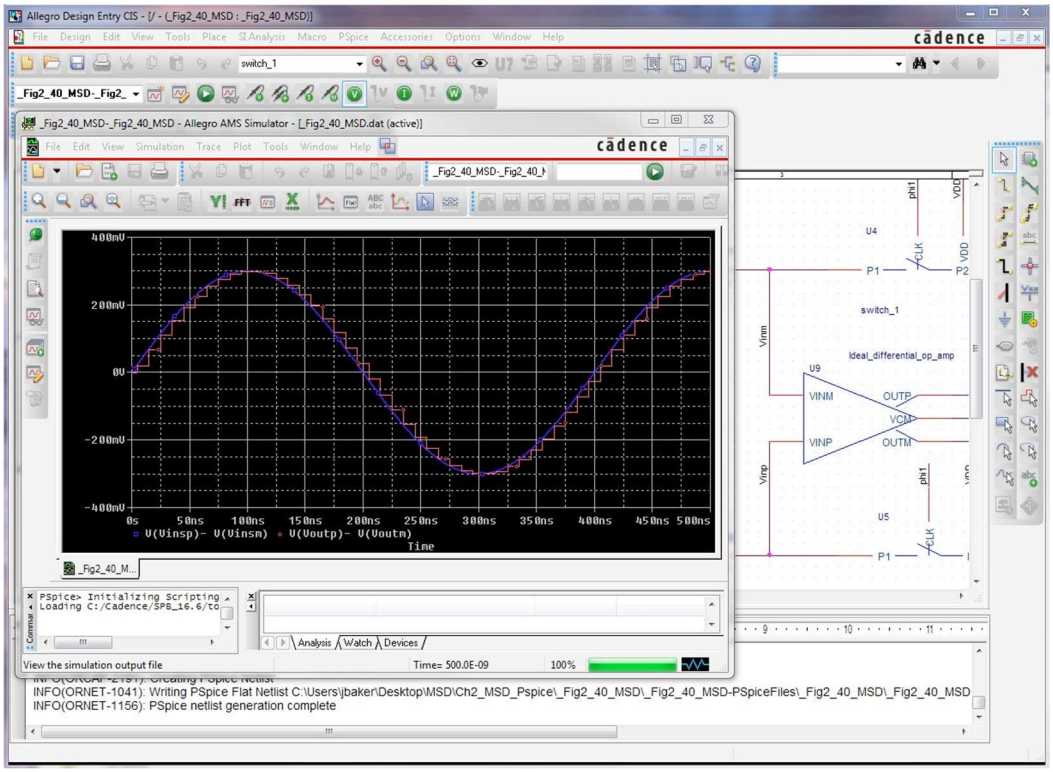


Figura 9 Exemplo da aplicação *PSpice* [13]

QUCS

O QUCS [14] é um simulador *Open Source* lançado em 2003 em permanente evolução quer através do contributo dos seus autores, quer por voluntários que procuram melhorar o software. Encontra-se disponível para sistemas *Linux*, *macOS* e *Windows*, e apresenta apresenta uma interface gráfica simples. Esta simplicidade permite a sua utilização por utilizadores menos experientes ou que iniciam o percurso académico. Possui várias bibliotecas e uma comunidade de desenvolvimento voluntário, o que permite o melhoramento e evolução do programa, seja a nível do interface ou das funcionalidades.

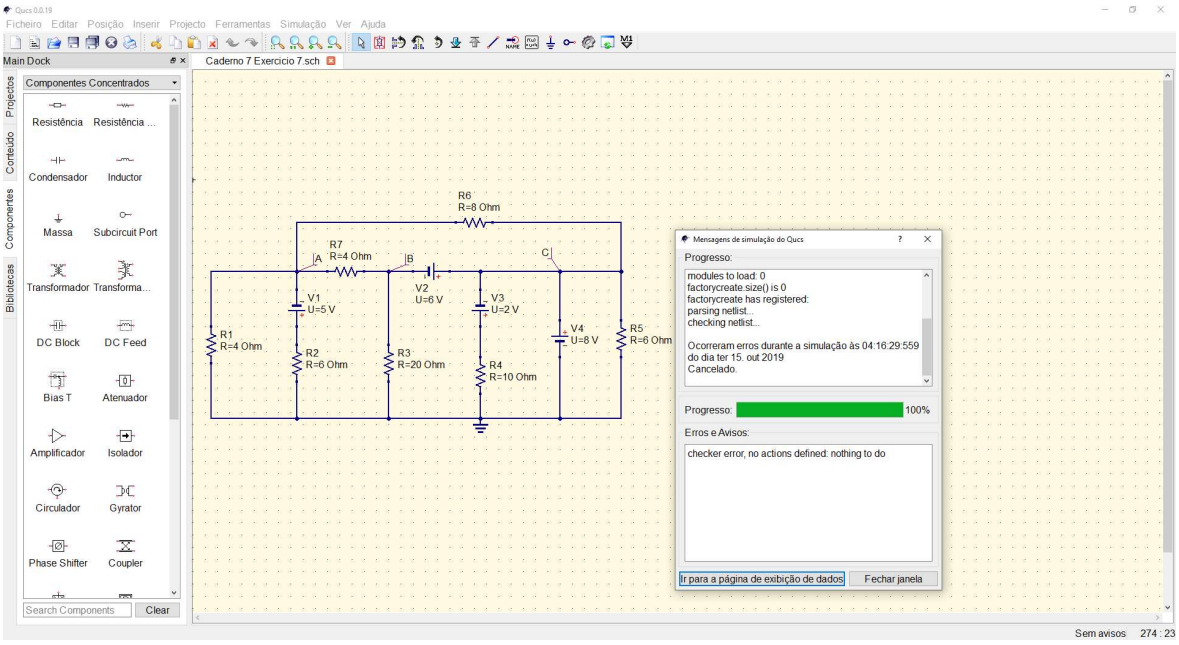


Figura 10 Exemplo da aplicação QUCS [14]

2.2. COMPARAÇÃO DE SIMULADORES

Após a descrição de alguns dos simuladores existentes, importa avaliar comparativamente os mesmos. Para os parâmetros de avaliação e atendendo ao âmbito académico em que se insere o presente trabalho, foi analisado o tipo de licença, o seu preço, os sistemas operativos que operam, a quantidade de bibliotecas e módulos disponíveis, a facilidade de adaptação à ferramenta, o suporte oficial, documentação e dimensão da comunidade.

Todas estas características foram avaliadas e sumarizadas na Tabela 2, assim como foi atribuída uma classificação final em função dos padrões avaliados.

Tabela 2 Tabela comparativa entre os simuladores de circuitos elétricos

	Licença	Preço	Plataforma	Bibliotecas	Facilidade de adaptação à ferramenta	Suporte e comunidade	Classificação global
CircuitLab	Software Proprietário	22,00 € / Ano	Browser	★★★	★★★★	★★★	★★
EasyEDA	Software Proprietário	Gratuito com Anúncios	Browser, Linux, macOS; Windows	★★★	★★★★	★★★	★★★
EveryCircuit	Software Proprietário	15,00 € Compra Única	Android, Google Chrome	★★★	★★★★	★★★★	★★★
KTechLab	GPL	Gratuito	Linux	★★★	★★★★	★★	★★★
LTspice	Freeware	Gratuito	macOS, Windows	★★★	★★	★★	★★
NI Multisim	Software Proprietário	685,00 € Compra Única	Windows	★★★★★	★★★	★★★★★	★★★
Oregano	GPL	Gratuito	Linux	★★★	★★★	★	★★
PSIM	Software Proprietário	Versão trial 30 dias	Windows	★★★	★★★	★★★★	★★
PSpice	Software Proprietário	6.000,00 € Compra Única	Windows	★★★★	★★	★★★★	★★
QUCS	GPL	Gratuito	Linux, macOS; Windows	★★★	★★★★★	★★★	★★★★

Das alternativas analisadas e tendo em conta o âmbito académico em que se insere o presente trabalho e bem assim o objectivo primordial de autoaprendizagem, facilmente concluímos que o QUCS é o mais completo em termos gerais, uma vez que se trata de um *Software open-source,* que se apresenta com um *software* apelativo, bibliotecas vastas e evolução constante.

3. MÉTODO DAS CORRENTES NOS RAMOS

O MCR é um método organizado que segue um conjunto de regras simples para a obtenção das equações do circuito. É baseado na Lei de Kirchoff das Malhas e dos Nós. A lei das Malhas (ou das tensões) dita que ao longo de qualquer Malha (percurso fechado sem repetição, ou seja, o seu trajeto só percorre os Nós uma vez) de um circuito, a soma algébrica das forças eletromotrizes é igual à soma algébrica das quedas de tensão nos componentes passivos. A lei de Kirchoff dos Nós (LKN) dita que a soma de todas as Correntes que fluem para o Nó é igual à soma das Correntes que saem do mesmo Nó, ou seja, em qualquer Nó do circuito a soma das correntes é nula.

3.1. MÉTODO PARA ANÁLISE MCR

Neste método as variáveis a determinar são as correntes nos ramos. As etapas a seguir para a execução técnica deste método são as seguintes [15]:

1. Recorrendo ao esquema do circuito, contar o número R de Ramos, o número N de Nós, o número de FC de Fontes de Corrente e calcular o número M = R – (N – 1) - FC de malhas linearmente independentes.

2. Atribuir a cada ramo uma corrente com um sentido arbitrário, indicando cada um no esquema, cujos valores (I_1 , I_2 , …, I_R) constituem as incógnitas a determinar.

3. Escrever as N-1 equações linearmente independentes referentes à lei dos Nós (desprezar um dos Nós), tendo em conta os sentidos arbitrados para as correntes no passo anterior como também o valor e o sentido da Corrente nos Ramos constituídos por Fonte de Corrente.

4. Escolher as malhas linearmente independentes de modo a que sejam incluídos todos os Ramos dos circuitos sem Fontes de Corrente, isto é, nenhum Ramo pode estar em falta na totalidade das Malhas selecionadas à exceção dos Ramos com Fonte Corrente. Escolher para cada malha um sentido de circulação. Escrever as M equações referentes às malhas escolhidas, tendo em conta os sentidos arbitrados, quer para a circulação de cada malha, quer para as correntes nos ramos que a constituem.

5. Resolver o sistema de R equações obtido (com N – 1 equações dos nós e M equações de malhas), calculando o calor das R incógnitas (I_1 , I_2 , …, I_R).

3.2. ANÁLISE DE CIRCUITO EXEMPLO

Para efeitos de demonstração deste método será analisado um exemplo de um circuito, representado na Figura 11, através do procedimento aqui apresentado.

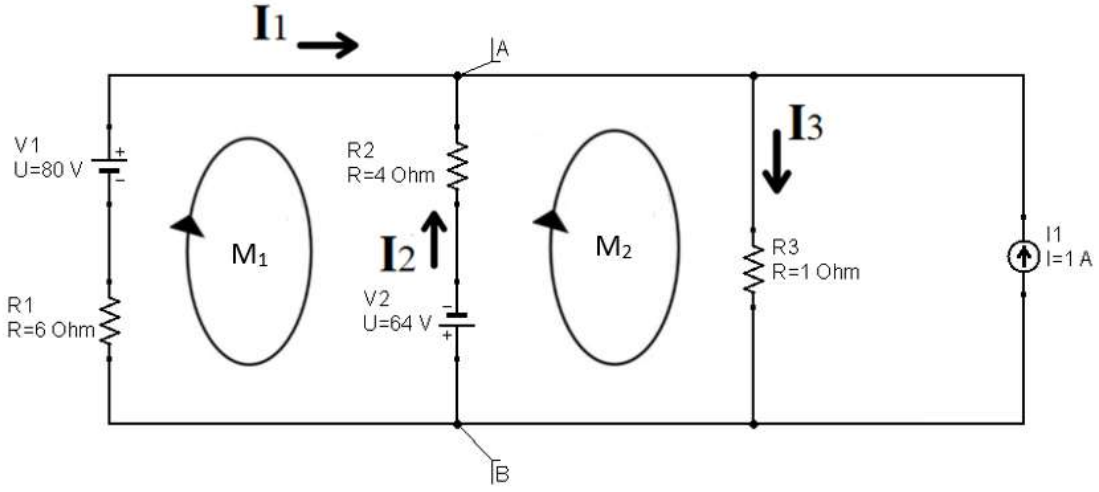


Figura 11 Exemplo de circuito para análise do MCR

- Passo 1

  Nós: A, B (2 Nós).

  Ramos: 4.

  Equações: $M = 4 - (2 - 1) - 1 = 2$ malhas linearmente independentes.

- Passo 2

  Para cada corrente de Ramo atribui-se um sentido, como indicado na Figura 11.

- Passo 3

  Escolhe-se, por exemplo, a equação do Nó A (despreza-se o nó B):

  $I_1 + I_2 + 1 = I_3$ (1)

- Passo 4

  Atribui-se um sentido de circulação para cada malha escolhida, como o indicado na figura.

  Estabelece-se as seguintes equações de malha:

  Malha 1: $6 \cdot I_1 - 4 \cdot I_2 = 64 + 80$ (2)

  Malha 2: $1.I_3 + 4 \cdot I_2 = -64$ (3)

- Passo 5

  Resolve-se o sistema:

$$\begin{cases} I1 + I2 + 1 = I3 \\ 6 \cdot I1 - 4 \cdot I2 = 64 + 80 \\ 1. I3 + 4 \cdot I2 = -64 \end{cases} \quad (4) \quad \text{cuja solução é} \quad \begin{cases} I1 \approx 14{,}53\ A \\ I2 \approx -15{,}71\ A \\ I3 \approx -1{,}18\ A \end{cases} \quad (5)$$

Na seleção das Malhas, torna-se fulcral garantir as seguintes condições:

- Todos os Ramos que constituem o circuito devem estar presentes em pelo menos uma Malha, pois só assim se torna possível obter um sistema de equações possível de se solucionar;
- Respeitar a construção de um sistema de R-FC equações (linearmente independentes) composto por R-FC incógnitas, o que implica escolher sempre Malhas constituídas por Ramos sem Fontes de Corrente. Consequentemente, obter-se-á um sistema de equações reduzido, isto é, formado pelo número mínimo de equações necessárias para tornar o mesmo passível de se solucionar.

Note-se que os sentidos das correntes foram escolhidos arbitrariamente pois, à priori, desconhecem-se os seus sentidos (são as incógnitas). Quando se obtém um valor negativo para uma corrente (como é o caso de I_2 e I_3) isso significa que o seu sentido real é contrário ao arbitrado (no Passo 2).
Após se obter o valor das correntes é possível calcular as tensões nos vários elementos do circuito. Esta é sem dúvida uma metodologia de fácil interpretação e aplicação que permite ter uma noção do comportamento de cada elemento perante o circuito em geral.

4. APLICAÇÃO U=RISOLVE: MÓDULO DO MÉTODO DAS CORRENTES NOS RAMOS

A aplicação desenvolvida pretende acrescentar à plataforma U=RISOLVE a possibilidade da análise de circuitos elétricos através do Método das Correntes nos Ramos. Pretende-se que os novos utilizadores se sintam motivados à autoaprendizagem, procurando fornecer uma ferramenta simples, mas completa, desenhada de forma a que seja possível ao utilizador obter uma resolução rápida passo a passo de qualquer circuito. Neste capítulo descreve-se a aplicação desenvolvida, a forma de interação com o utilizador e as tecnologias que permitem obter a informação do circuito.

4.1. OBJETIVOS E ARQUITETURA

A plataforma U=RISOLVE é uma aplicação totalmente desenvolvida em JavaScript com interface gráfica desenvolvida em HTML5 e CSS. Essa aplicação foi o resultado de um outro projeto elaborado no âmbito da disciplina de PESTA[1], onde se desenvolveu a funcionalidade da Análise de circuitos pelo Método das Tensões nos Nós. Pretende-se com o presente trabalho, adicionar mais uma funcionalidade à plataforma, que, no caso deste projeto, será o MCR. Passa a ser possível aos utilizadores terem acesso às equações dos Nós e das Malhas do circuito elétrico bem como foi conseguido, como um extra aos objetivos inicialmente estipulados, o valor das Correntes em cada Ramo. O processo pode ser descrito em três passos, como também é visualmente explicado através da Figura 12:

1. Criar uma *netlist* para o circuito simulado no QUCS, quer através do menu de simulação quer pressionando a tecla F6;
2. Guardar a *netlist* em formato de texto no diretório escolhido pelo utilizador;
3. Fazer *upload* da *netlist* gerada no QUCS para a plataforma U=RISOLVE e carregar no botão "Analisar Circuito"

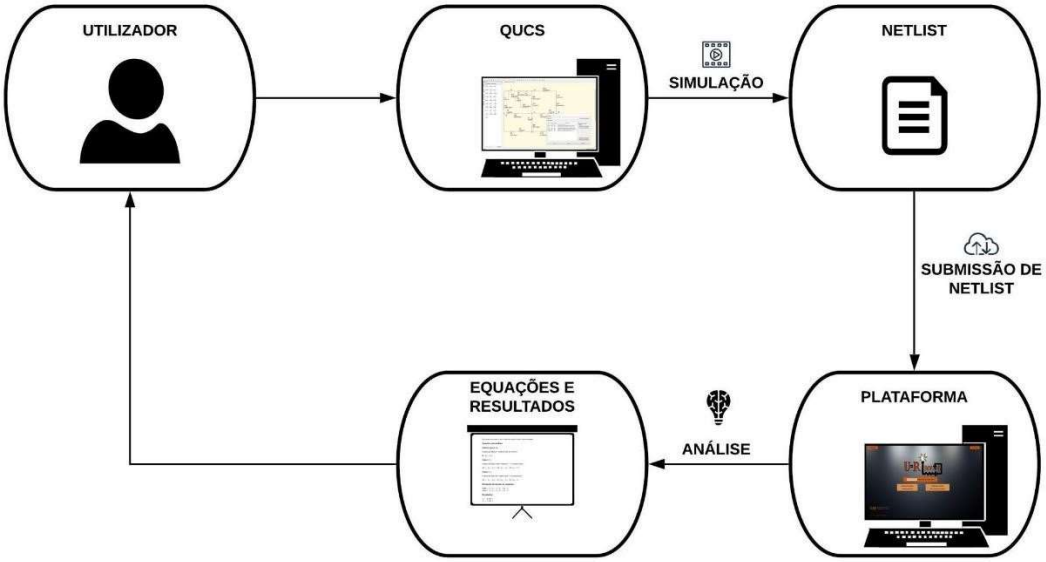


Figura 12 Lógica de funcionamento da plataforma U=RIsolve

Após o upload, os resultados são apresentados através duma janela *pop-up* que se abre no browser; sempre que o utilizador pressione um dos botões das diferentes análises disponíveis e que tenha feito o upload da *netlist* é aberta uma janela *pop-up*. Assim possibilita-se ao utilizador analisar diferentes circuitos sem ser necessário abrir uma nova janela da aplicação ou extrair os resultados das simulações anteriores. O resultado da análise é exibido por etapas, possibilitando assim que o utilizador acompanhe todos os passos, promovendo a sua autoaprendizagem.

O resultado é apresentado pela seguinte ordem:

1. Inicialmente são apresentadas as características do circuito em análise, indicando-se o número de Ramos, Nós e Fontes de Corrente, seguido do número de Malhas possíveis e Malhas necessárias para a resolução, com apresentação da fórmula;
2. De seguida são apresentadas as Correntes de cada Ramo identificadas por um índice numérico, bem como o sentido arbitrado para cada uma delas que é sinalizado indicando os nós que delimitam cada Ramo. Também são mostrados os componentes constituintes de cada ramo pela ordem de fluxo da corrente.
3. Posteriormente, são apresentadas as equações dos Nós em que, por definição da aplicação, as Correntes são escritas no primeiro membro da equação; as Correntes que convergem para o Nó são antecedidas de um sinal positivo; as Correntes que divergem do Nó são antecedidas de um sinal negativo;
4. Seguidamente são apresentadas as equações das Malhas. Inicialmente, para facilitar a compreensão do utilizador, é apresentada o sentido da Corrente de Malha indicando em que Nó e componente começa a leitura seguido dos restantes componentes. Depois é apresentada a equação propriamente dita em que, por definição da aplicação, são apresentadas as quedas de tensão no primeiro membro da equação e as forças eletromotrizes no segundo.
5. Posteriormente, no caso da análise em Corrente Contínua, o sistema de equações surge com as Correntes dos Ramos como únicas incógnitas, tendo substituído nas equações produzidas anteriormente os valores conhecidos de componentes. Em circuitos em Corrente Alternada, este ponto não se executa devido à complexidade da operação.
6. No final, em Corrente Contínua, são mostrados os resultados das Correntes nos Ramos.

Importa sublinhar que neste módulo da aplicação as malhas escolhidas aleatoriamente pelo *script* nunca vão conter Fontes de Corrente de forma a respeitar o Método das Correntes nos Ramos. Ao escolher apenas Malhas sem Fontes de Corrente reduz-se o sistema ao número mínimo de equações, tornando mais simples a análise ao circuito e diminuindo o tempo de processamento do algoritmo.

Também importa referir que o utilizador deve identificar os nós do circuito; não o fazendo, o programa QUCS atribui uma designação a cada Nó "netX", onde "X" representa um número de identificação que é incrementado sempre que é detetado um novo Nó virtual. Esta

atribuição por parte do programa quando gera a netlist pode tornar confusa a análise das equações, tornando de difícil identificação os Nós que estariam incluídos nos potenciais das equações geradas. Assim sugere-se que os utilizadores identifiquem todos os nós reais e os nomeiem facilitando a interpretação dos resultados, o que contribui para a autoaprendizagem.

A aplicação contém igualmente uma página de instruções para tornar a utilização da mesma mais fácil e acessível, e bem assim exemplos de circuitos já matematicamente resolvidos e simulados, em conjunto com a *netlist* respectiva. Os exemplos encontram-se divididos por Corrente Contínua e Corrente Alternada e também por nível de complexidade.

4.2. ALGORITMO DE ANÁLISE

A estruturação do algoritmo tem por base o trabalho já realizado para a análise MTN [1] bem como o trabalho realizado para a análise MCM, possuindo operações iniciais idênticas, independentemente dos métodos serem diferentes. No entanto, procurou-se adaptar e melhorar etapas referentes à primeira versão da aplicação, que continha apenas o módulo para análise MTN.

O algoritmo de análise desenvolvido possui seis etapas:

- Filtragem e armazenamento de informação da *netlist* numa matriz;
- Remoção de aparelhos de medição;
- Deteção de erros na informação tratada;
- Definição de Nós reais e Ramos;
- Geração de combinações de Ramos;
- Identificação e seleção de Malhas;

A divisão por etapas, permitiu por um lado a organização de todo o processo com objectivos claros e sequenciais. Assim optimizou-se quer a organização e a informação bem como permitiu um tratamento sequencial da informação, evitando falhas posteriores, avançando assim a análise.

De seguida abordaremos em pormenor todas as fases do algoritmo de análise. A Figura 13 representa o fluxograma de funcionamento do algoritmo.

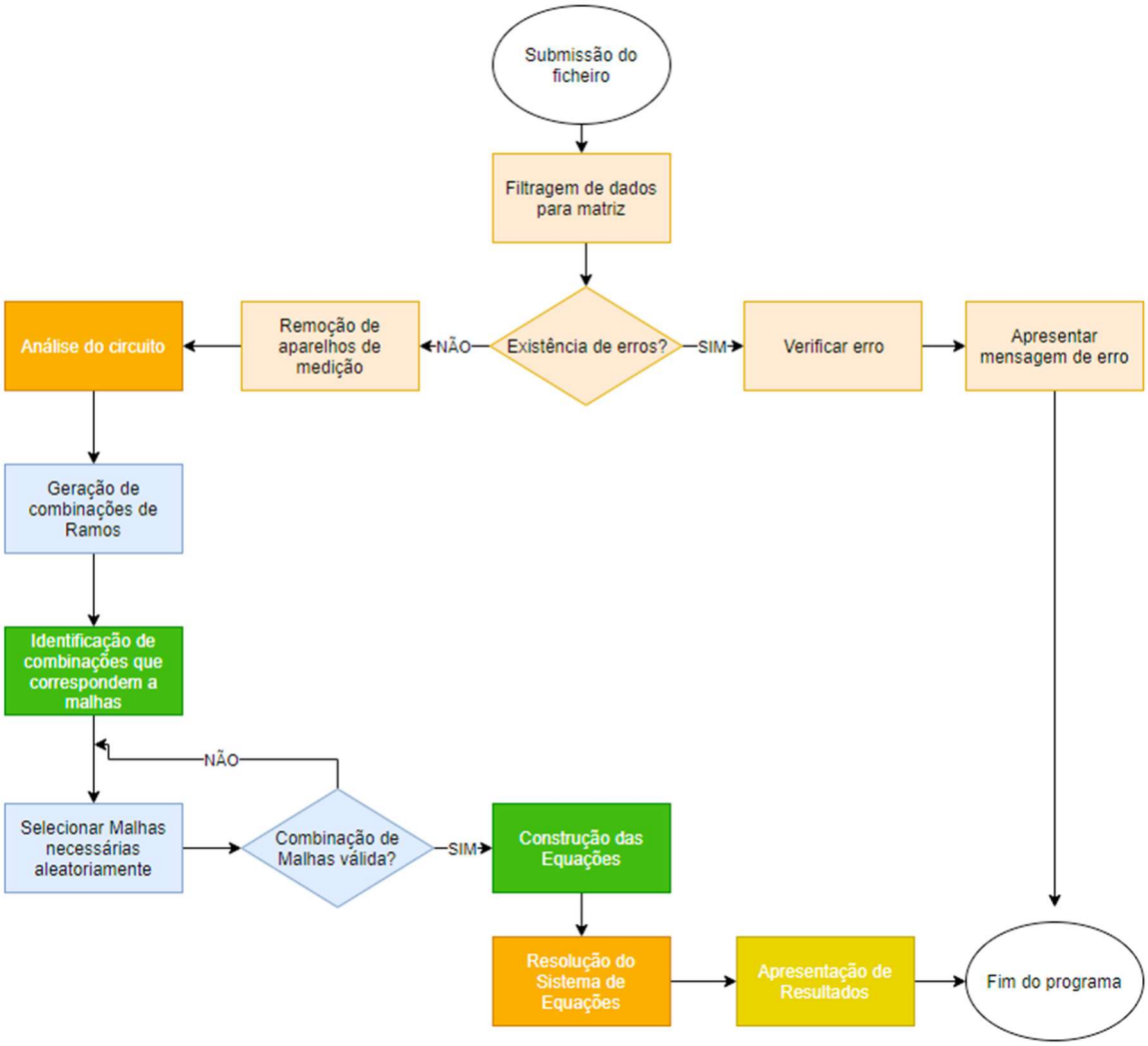


Figura 13 Fluxograma da aplicação U=RISOLVE para o MCR

Todas as contribuições no desenvolvimento de todo o módulo de análise MCR estão discriminadas na Figura 14.

Figura 14 Processos ordenados com menção das contribuições

4.2.1. FILTRAGEM E ARMAZENAMENTO DE INFORMAÇÃO

Num primeiro momento, o algoritmo extrai toda a informação relevante que consta na *netlist* gerada pelo QUCS [16], procurando todos os componentes existentes (Fontes de Tensão, resistências, condensadores, bobines ou Fontes de Corrente) agrupando-os. Dito de outra forma, a aplicação primeiramente armazena numa *string* todos os caracteres da *netlist* submetida, posteriormente procura por todos os componentes do circuito, os Nós virtuais de cada um e os seus respetivos valores. Na Figura 16 está representada uma matriz de dados gerada a partir de uma *netlist,* representada na Figura 15.

```
# Qucs 0.0.19  C:/Users/PC/Desktop/Circuitos Capitulo 5/DC/Nivel 2/Caderno 7 Exercicio 7.sch

Vdc:V4 _C _gnd U="8 V"
Vdc:V2 _C _B U="6 V"
Vdc:V3 _net1 _C U="2 V"
Vdc:V1 _net0 _A U="5 V"
R:R1 _gnd _A R="4 Ohm" Temp="26.85" Tc1="0.0" Tc2="0.0" Tnom="26.85"
R:R2 _gnd _net0 R="6 Ohm" Temp="26.85" Tc1="0.0" Tc2="0.0" Tnom="26.85"
R:R7 _B _A R="4 Ohm" Temp="26.85" Tc1="0.0" Tc2="0.0" Tnom="26.85"
R:R6 _C _A R="8 Ohm" Temp="26.85" Tc1="0.0" Tc2="0.0" Tnom="26.85"
R:R3 _gnd _B R="20 Ohm" Temp="26.85" Tc1="0.0" Tc2="0.0" Tnom="26.85"
R:R4 _gnd _net1 R="10 Ohm" Temp="26.85" Tc1="0.0" Tc2="0.0" Tnom="26.85"
R:R5 _gnd _C R="6 Ohm" Temp="26.85" Tc1="0.0" Tc2="0.0" Tnom="26.85"
```

Figura 15 Exemplo de *netlist*

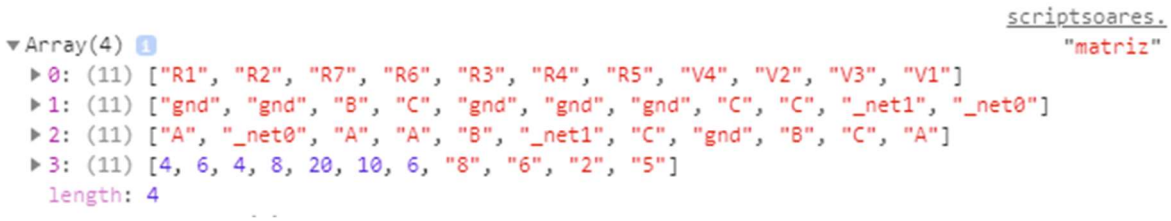


Figura 16 Filtragem de informação da *netlist* para uma matriz

O QUCS quando gera a *netlist* identifica de forma distinta cada componente, seguindo sempre o mesmo padrão. No caso da identificação dos componentes, o *script* pesquisa pelo excerto "R:Ri" para definir uma resistência com índice "i", seguidamente procura e armazena os Nós virtuais do mesmo, isto é, o ponto de ligação dos terminais do mesmo. No fim é armazenado o valor da resistência. O mesmo acontece com os restantes componentes definidos pelas respetivas sequências de carateres:

- “C:C” para condensadores;
- “L:L” para bobines;
- “Vdc:V” para Fontes de Tensão DC;
- “Vac:V” para Fontes de Tensão AC;
- “Idc:I” para Fontes de Corrente DC;
- “Iac:I” para Fontes de Corrente AC;
- “IPro:” para amperímetros.

Os voltímetros podem ser igualmente encontrados por uma sequência de carateres, porém a sua aquisição seria inútil neste sentido, pois em nada interferem na estrutura do circuito.

Depois, torna-se importante confirmar se os valores guardados estão na escala correta.

Nesta fase, o algoritmo procura ainda a presença de prefixos do Sistema Internacional (SI) após ler o valor do componente. Tais prefixos são aceites pelo QUCS no momento que são definidos os valores, como se pode observar na Figura 17.

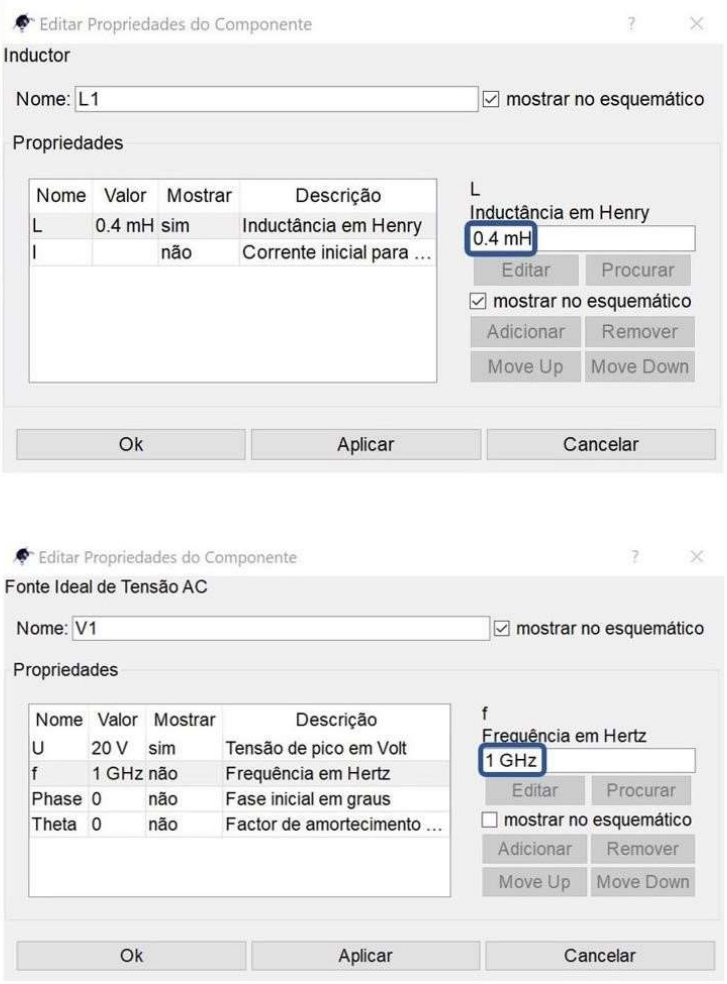


Figura 17 Exemplo de definição de valores no QUCS

Por fim, o *script* faz a conversão dos valores para a unidade SI, multiplicando por 10 elevado ao valor do prefixo. Este ponto revela ser importante nesta fase para simplificar a manipulação das equações na parte final do algoritmo.

4.2.2. MÉTODO DE AQUISIÇÃO DE NÓS E RAMOS

A aquisição dos Nós e dos Ramos com base na matriz estabelecida é dos pontos mais importantes em todo o processo de análise do circuito. A partir deste ponto, será possível estabelecer uma relação de ligação entre os Nós reais e os Ramos do circuito. Esta correspondência será necessária para, mais à frente, o *script* poder identificar as Malhas existentes no circuito sem ter que recorrer ao esquemático gráfico do circuito. De seguida apresenta-se, na Figura 19, a matriz de dados obtida pela *netlist* associada ao circuito exemplo representado pela Figura 18.

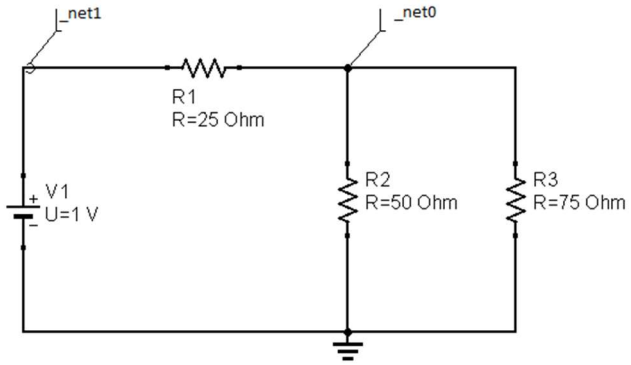


Figura 18 Exemplo de circuito simples elaborado no QUCS

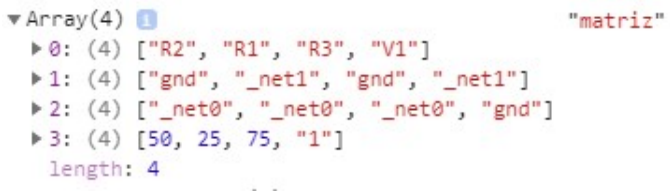


Figura 19 Matriz com valores obtidos a partir da *netlist* do circuito da Figura 15

Esta matriz define univocamente o circuito exemplo. Depois de todas as informações devidamente filtradas e organizadas, outras características serão também relevantes, como é o caso dos Nós reais e dos Ramos do circuito. De seguida é apresentada uma breve descrição da aquisição destes elementos, tendo como base a matriz da Figura 19:

- **Nós virtuais** – Um Nó virtual representa o intervalo onde são ligados apenas dois componentes e é gerado automaticamente pelo simulador assim que se interligam dois elementos. Desta forma, só se poderá verificar a presença do nó virtual na matriz duas vezes. Neste caso, só o ponto "_net1" se considera com um Nó virtual;
- **Nós Reais** – Um Nó real é o ponto em que se ligam no mínimo três componentes, ou seja, os Nós reais devem constar três ou mais vezes na matriz. Verifica-se que, para o circuito exemplo, "_net0" (liga R1,R2 e R3) e "gnd" (liga R2,R3 E V1) são Nós reais. O método de aquisição destes Nós é representada pelo fluxograma da Figura 20.

- **Ramos** – Um Ramo de um circuito é o percurso entre dois Nós reais consecutivos. Só após a aquisição dos Nós reais será possível definir os Ramos do circuito. Na matriz, define-se como ponto de partida um componente que esteja ligado a um Nó real e analisa-se o caminho que este deve percorrer pelo outro Nó, até encontrar outro Nó real. No exemplo, começando por V1, verifica-se que este está ligado a um dos Nós reais ("gnd"), portanto, testará se o componente seguido do seu Nó virtual está ligado a um Nó real, neste caso R1, que por sua vez conecta com o Nó real "_net0". Os outros dois Ramos são mais simples de se definirem dado que cada é formado apenas por um componente, portanto, ambos os pontos de ligação desses elementos na matriz correspondem a Nós reais. Este raciocínio é demonstrado pelo fluxograma da Figura 21.

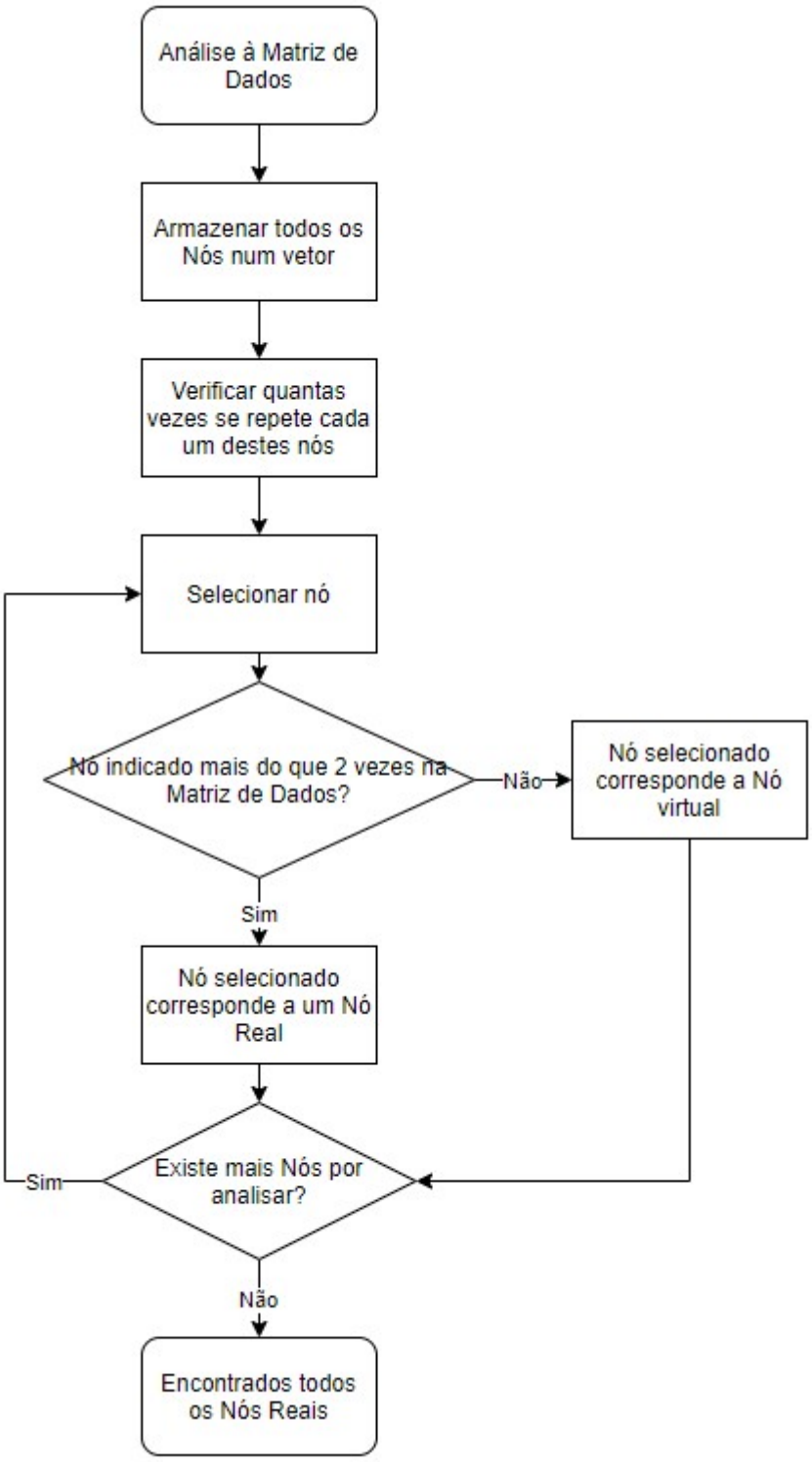


Figura 20 Fluxograma do processo de aquisição de Nós reais a partir da Matriz de Dados

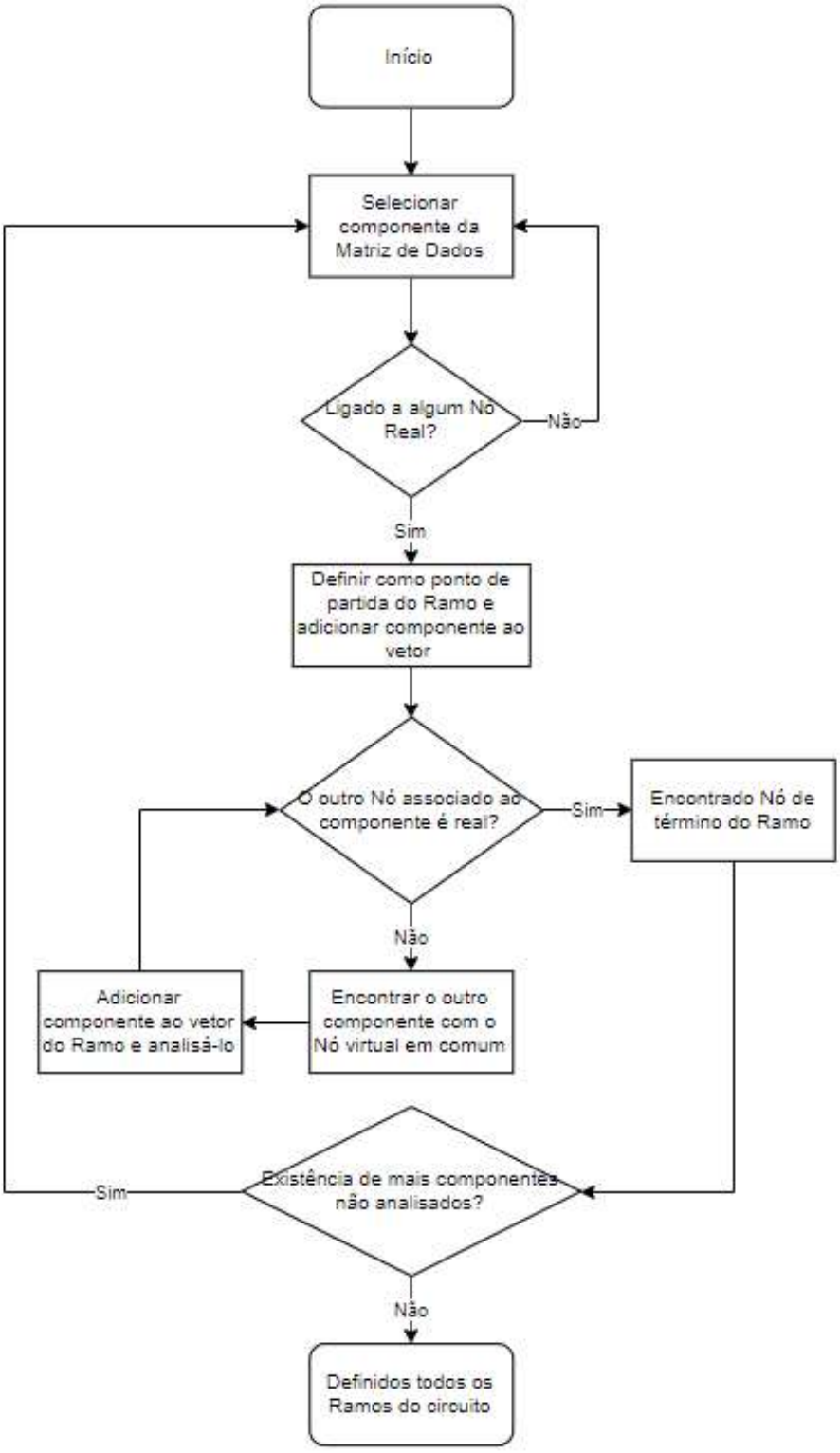


Figura 21 Fluxograma do processo de aquisição de Ramos a partir da Matriz de Dados

4.2.3. REMOÇÃO DE APARELHOS DE MEDIÇÃO

Na aquisição de informação acerca da composição do circuito a partir da *netlist* é natural que, sempre que utilizador integrar no circuito instrumentos de medição, estes constem na matriz de dados inicial. Como estes elementos são irrelevantes para a análise do circuito, torna-se necessário retirar da matriz estes aparelhos.

No que toca aos voltímetros, estes não são um obstáculo dado que não modificam qualquer propriedade do circuito: Por serem ligados em paralelo, basta apenas não se armazenar qualquer informação sobres estes, como foi anteriormente mencionado.

No que diz respeito aos amperímetros, dado o facto de estes elementos serem ligados em série, irão criar obrigatoriamente um Nó virtual. Isto é, se os amperímetros passassem pelo mesmo processo de filtragem que os voltímetros, o circuito ficaria aberto no Ramo em que o amperímetro fizesse parte. Uma forma de solucionar este problema será armazenar os dados dos amperímetros e curto-circuitar os seus Nós. De seguida é apresentado um fluxograma na Figura 22 que explica a lógica aplicada.

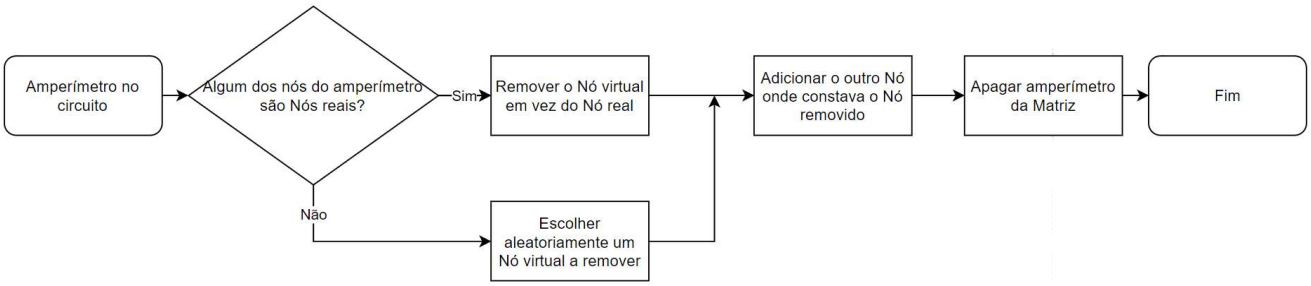


Figura 22 Fluxograma do método de remoção de amperímetros

A aplicação deste método traz uma grande vantagem do ponto de vista de quem está a usar a aplicação. O utilizador não terá a necessidade de apagar os instrumentos de medição de simulação, tornando a aplicação mais flexível em termos de utilização.

4.2.4. DETEÇÃO DE ERROS

O utilizador precisa de seguir determinadas regras quando está a elaborar o circuito elétrico no QUCS. Estas regras implicam que o utilizador consiga identificar e referenciar os Nós reais, conhecimento esse essencial a qualquer aluno no âmbito da EE. Assim torna-se necessário que o utilizador:

- Identifique todos os Nós reais com uma letra;
- Identifique o Nó de referência com o símbolo de *ground* existente no QUCS.

Caso o utilizador não respeite estas regras, estas serão evidenciadas no momento de filtragem da informação que se obtém através da *netlist*. Em face dessa verificação e detetando-se esses erros, aparecerá ao utilizador uma mensagem de erro e, bem assim, a instrução que deverá seguir para os corrigir. Para expor estes erros ao utilizador, recorreu-se a uma função do *JavaScript* designado por *alert(),* que ao ser invocada, faz aparecer uma caixa de alerta com uma mensagem. Essa mensagem corresponde a uma *string* dinâmica que no final de cada verificação vai acrescentando o texto referente ao erro que foi detetado.

Tal como se disse, essa caixa de alerta menciona, para além dos erros detetados, as instruções para os corrigir, tal como se pode ver na Figura 23.

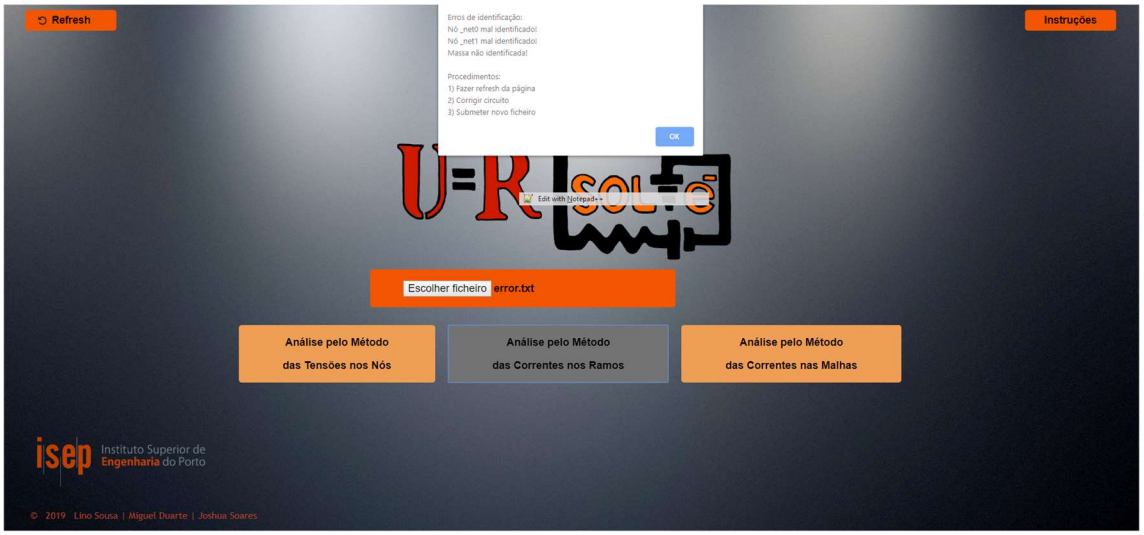


Figura 23 Exemplo de alerta de erro

4.2.5. GERAÇÃO DE COMBINAÇÕES DE MALHAS

O emprego de um algoritmo que gerasse combinações de Malhas surge da necessidade de garantir que o método de análise do algoritmo seja robusto e infalível. Só assim é assegurado que a aplicação devolva os resultados corretos sempre que é pedido, analisando todas as hipóteses de Malhas possíveis.

O objetivo será criar combinações de todos os Ramos do circuito que não sejam constituídos por Fontes de Corrente, sendo que cada Ramo é identificado numericamente. É de frisar que na criação de combinações, os conjuntos deverão ser compostos por, no mínimo, dois Ramos e, no máximo, o número equivalente ao número de Nós Reais do circuito. Para facilitar a interpretação da problemática, tem-se representado na Figura 24 uma simplificação de um circuito composto por 4 Nós e 9 Ramos, onde será possível criar no máximo combinações de 4 Ramos. De seguida é apresentado na Figura 25 parte das combinações geradas.

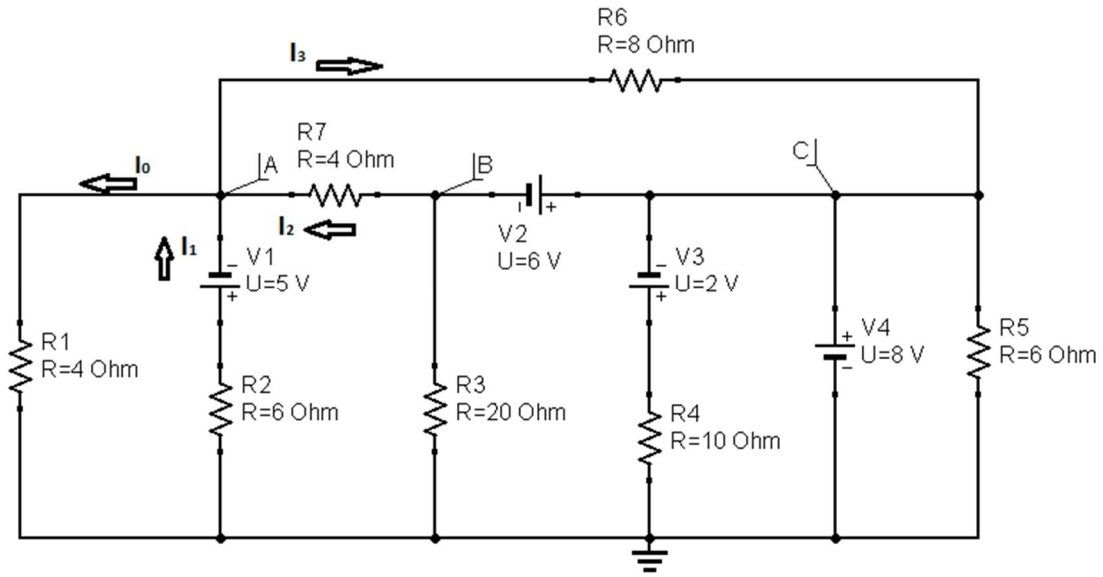


Figura 24 Simplificação de um circuito exemplo

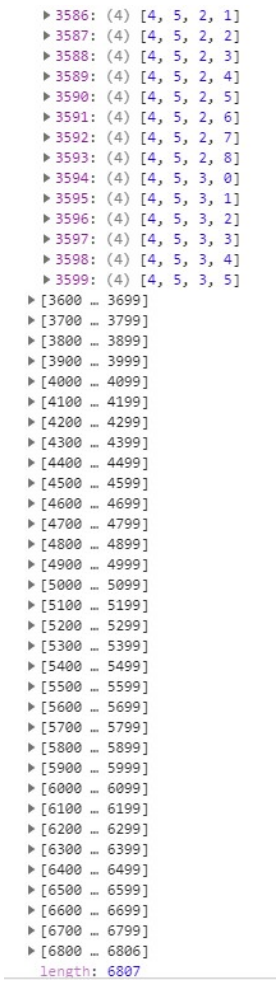


Figura 26 Exemplo de combinações de Ramos

Como se pode observar pela Figura 26, onde é apresentada parte de uma lista de vetores , chegou-se a um total de 6807 combinações e arranjos diferentes de Ramos que serão analisadas e selecionadas na fase seguinte.

Este valor é o resultado da soma algébrica de: $C_2^9 + C_3^9 + 9^4 = \frac{9!}{2!(9-2)!} + \frac{9!}{3!(9-3)!} + 9^4$, em que a primeira adição trata-se da soma de combinações de 9 Ramos, 2 a 2, com combinações de 9 Ramos, 3 a 3. Isto porque a disposição dos Ramos é irrelevante quando se trata de Malhas de 2 ou 3 Ramos. Seguidamente adiciona-se um arranjo completo de 9 Ramos, 4 a 4. Neste ponto, o algoritmo de geração de arranjos não se encontra otimizado pois, como se tratam de arranjos completos, existe repetição de Ramos no mesmo conjunto. No entanto,

isto não impede o algoritmo de achar todas as possibilidades de Malhas possíveis para o circuito a ser analisado.

O algoritmo a que se recorreu de modo a produzir este resultado baseia-se no algoritmo geral para Javascript das combinações [17].

Numa fase de refinação, constatou-se que o facto de o algoritmo criar um número grande de combinações, causado não só pelo elevado número de Nós reais e Ramos mas também pela disposição destes na construção do circuito, poderia levar a um *overflow* da *stack* do *browser*, impossibilitando o *script* de prosseguir a análise. Por isto, decidiu-se adaptar o algoritmo implementando novas rotinas de modo a simplificar o processo e torná-lo mais leve para o processador. As novas implementações passam pelos seguintes pontos:

- No caso concreto de conjuntos de 2 ou 3 Ramos, são geradas combinações, pois neste caso a ordem da disposição é irrelevante. Pode-se verificar anteriormente na Figura 22(figura do circuito) que, em termos de análise ao circuito e identificação das Malhas, a Malha {4,7,8} é igual à {4,8,7} ou à {8,7,4};

- Se o circuito for constituído por mais de 6 Nós e 6 Ramos, o algoritmo apenas gera as combinações de todos os Ramos sem repetição em conjuntos máximos de 6 Ramos. Esta implementação limita o método à análise de Malhas constituídas até 6 Ramos no máximo, excluindo possíveis Malhas de 7 ou mais Ramos.

  Estas combinações e permutações são armazenadas num vetor que será testado posteriormente, um a um, na etapa seguinte.

4.2.6. IDENTIFICAÇÃO E SELEÇÃO DE MALHAS

No desenvolvimento desta parte do algoritmo, chegou-se à descoberta de noções importantes que dizem respeito à composição de Malhas de um circuito elétrico.

Definiram-se algumas condições necessárias à identificação de uma Malha. Esta tem de ser composta por no mínimo dois Ramos, sendo que estes se encontram ligados pelos mesmos Nós reais, e no máximo "N" Ramos, sendo que "N" corresponde ao número de Nós reais do circuito. Nunca o número de Nós reais de uma Malha poderá exceder o número de Nós reais do circuito pois, neste caso, significaria que se estaria obrigatoriamente a repetir um Nó real. Após a geração de todas as combinações de Ramos, torna-se necessário identificar quais correspondem a Malhas do circuito. A identificação de Malhas passa por uma rotina contínua de análise de cada combinação, verificando primeiramente o seu tamanho, ou seja: o número de Ramos que constituem essa combinação. Concluindo, o *script* identifica as Malhas por ordem crescente de tamanho, começando pelas Malhas com dois Ramos até Malhas com seis Ramos.

Começando pelo caso mais simples, uma combinação composta apenas por dois Ramos, o algoritmo tem somente que verificar se os dois Nós reais dos dois Ramos constituintes da Malha são os mesmos, independentemente da ordem. Como se verifica na Figura 27, ambos os Ramos partilham dos mesmos Nós reais. Através do fluxograma da Figura 28, será possível visualizar a lógica inerente ao processo de identificação de Malhas de dois ou mais Ramos.

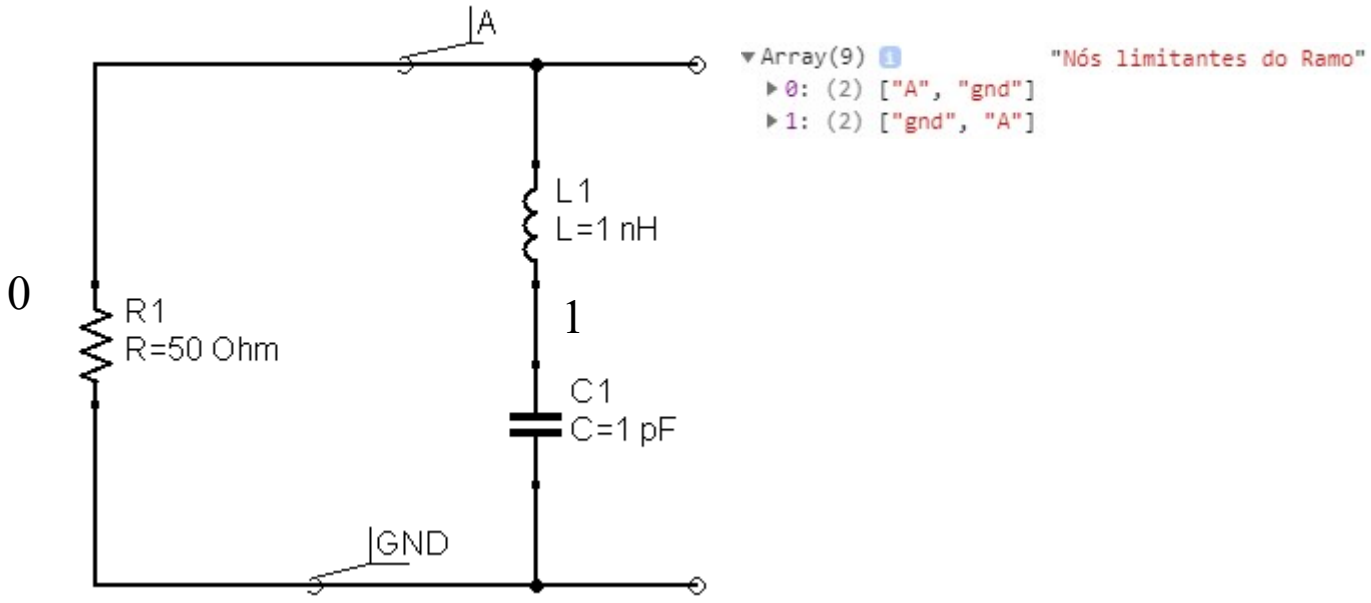


Figura 27 Exemplo de Malha com dois Ramos

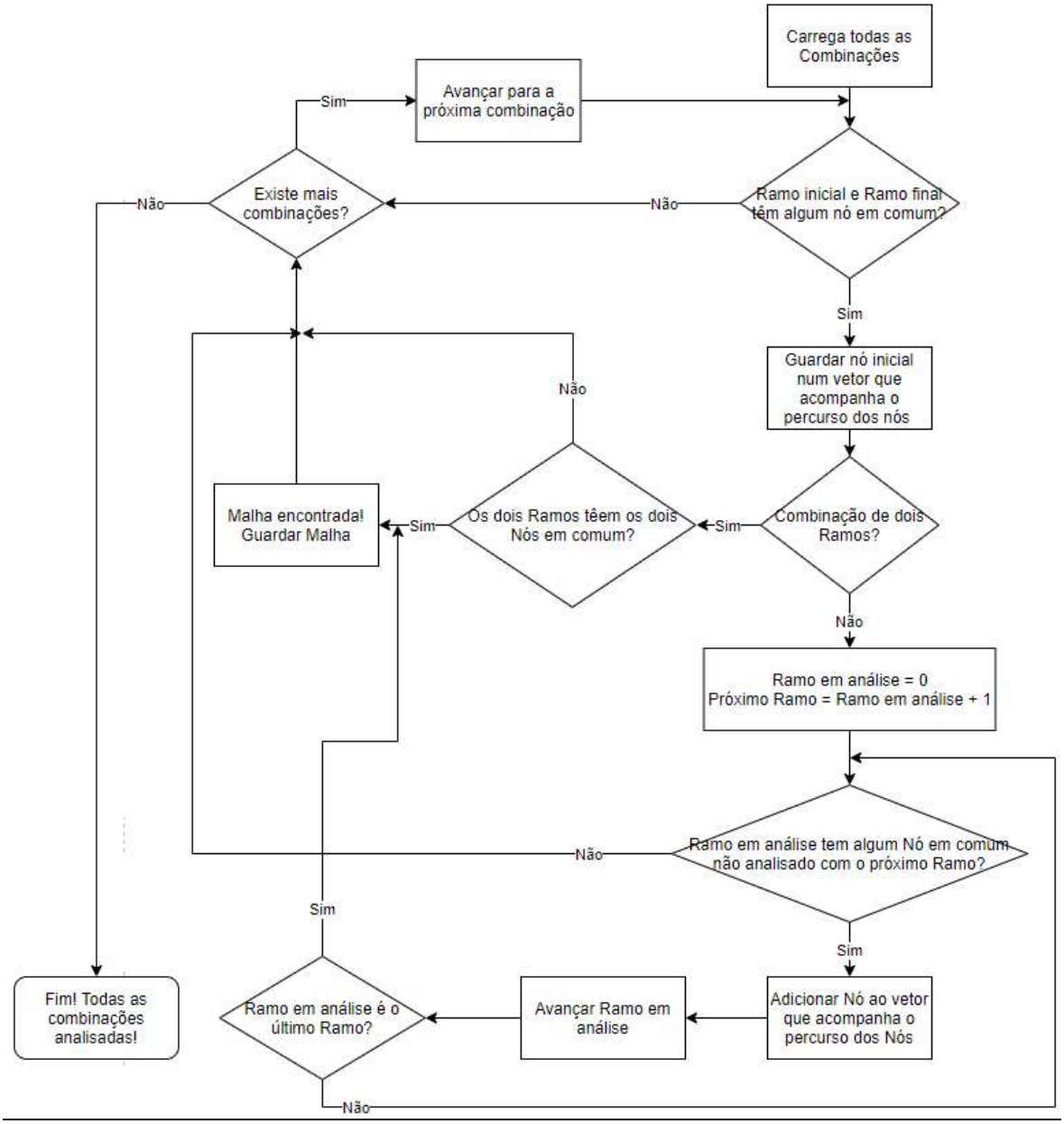


Figura 28 Fluxograma do algoritmo de identificação de Malhas

Depois de se identificar as Malhas mais simples procede-se à análise de combinações de três e/ou mais Ramos. Neste ponto, o algoritmo analisa tanto os Nós do primeiro Ramo como os do segundo, mas só prossegue a validação se e só se um dos Nós reais for comum aos dois Ramos. Esta condição é necessária pois caso não se verifique, significa que não existe nenhuma ligação entre os dois Ramos. No caso de ambos os Nós reais serem comuns aos dois Ramos trata-se de uma malha já identificada anteriormente, uma Malha de dois Ramos. O *script* ao identificar este primeiro Nó real comum, armazena-o na segunda posição de um vetor que acompanha o percurso percorrido pela Malha, sendo que a primeira posição é ocupada pelo outro Nó real do

primeiro Ramo, Ramo este que ainda não foi correspondido. A partir deste ponto o algoritmo percorre o resto da combinação, comparando o segundo Ramo com o terceiro, e assim sucessivamente. Quando se tiver verificado todas as ligações entre os Ramos intermédios, e se estiver analisar o último Ramo da combinação, o algoritmo verifica se o primeiro Nó real armazenado no vetor de Nós corresponde ao Nó real do último Ramo. Se esta condição se verificar, foi encontrada uma combinação que corresponde a uma Malha do circuito.

Depois de identificar todas as combinações de Ramos sem Fontes de Corrente geradas que correspondem a malhas do circuito torna-se necessário eliminar as Malhas duplicadas. O script apaga todas as Malhas que se repetem mas numa ordem diferente, isto é, Malhas com os mesmos Ramos mas com sentido contrário. Com isto assegura-se que o script não utilizará, posteriormente na análise do MCR, a mesma Malha duas vezes pois isto levaria à irresolução do problema. Esta filtragem também garante que se obtenha o número máximo de Malhas sem fontes de corrente que o circuito submetido poderá ter. É de frisar que este valor não é obtido por nenhuma fórmula matemática pois dois circuitos com o mesmo número de Nós reais e Ramos pode ter um número máximo de Malhas possíveis diferente.

Depois de se obter todas as malhas (sem fontes de corrente) possíveis do circuito, procede-se à seleção das necessárias para a resolução do método. Durante este processo definiram-se algumas condições de modo a obter o conjunto de Malhas mais conveniente à análise. Sempre que é escolhida uma malha, esta é guardada num vetor de Malhas selecionadas e apagada do vetor de Malhas possíveis. O script começa por selecionar Malhas que obedeçam às seguintes condições obrigatórias:

- Todos os Ramos do circuito devem ser incluídos em pelo menos uma Malha;
- O número total de Malhas selecionadas deve ser igual ao número de Malhas.

Se alguma destas condições não se verificar, o script executa uma função de *shuffle* no vetor de Malhas possíveis, voltando a selecionar aleatoriamente e a testar novamente as condições. Quando todas a condições se verificarem, o vetor Malhas selecionadas fica preenchido, isto é, o script já detém o número de malhas suficientes para poder aplicar o MCR. Assim sendo, o programa avança para a próxima etapa.

4.3. CONSTRUÇÃO DAS EQUAÇÕES

A geração das equações em código requer um elevado grau de clareza e objetividade. Com o propósito de melhorar o processo de leitura e interpretação, os resultados são apresentados com um nível crescente de complexidade. Em primeiro lugar começa-se por apresentar os componentes integrantes de cada Ramo e por arbitrar o sentido da sua corrente. O *script* atribui o sentido da corrente baseando-se no vetor nos_limitantes, vetor este que é constituído pelos nós que delimitam cada Ramo. Na Figura 30 é apresentado um conjunto destes vetores, que condiz com o circuito da Figura 29 que tem vindo a servir de exemplo durante este relatório.

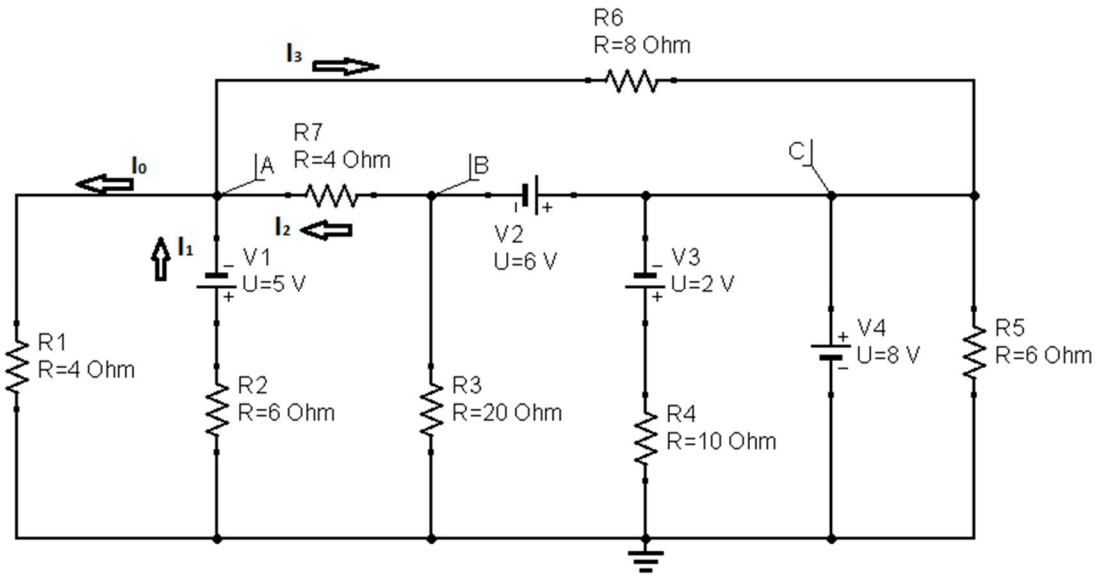


Figura 29 Circuito exemplo

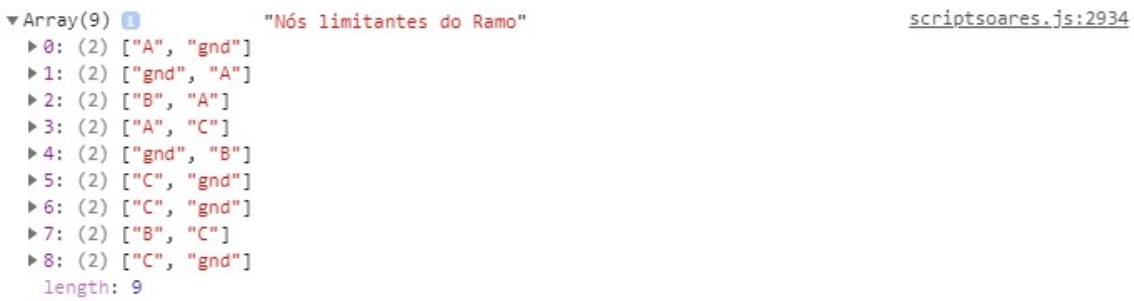


Figura 30 Vetores Nós limitantes do Ramo

O sentido da Corrente é atribuído seguindo sempre a ordem dos nós constituintes do vetor do respetivo Ramo, isto é, a Corrente diverge do primeiro Nó do vetor convergindo para o segundo nó do mesmo. A ordem da apresentação dos componentes é feita seguindo o sentido da Corrente de modo a facilitar a interpretação do circuito por parte do utilizador, ou seja, o primeiro componente a ser mostrado é o que se encontra mais próximo do Nó de onde Corrente diverge e o último é o que se encontra mais próximo do Nó para onde a Corrente converge. Na Figura 31 é apresentado um conjunto de vetores componentes_ramo onde são guardados os componentes constituintes de cada Ramo, onde cada componente já se encontra ordenado segundo o sentido da Corrente.

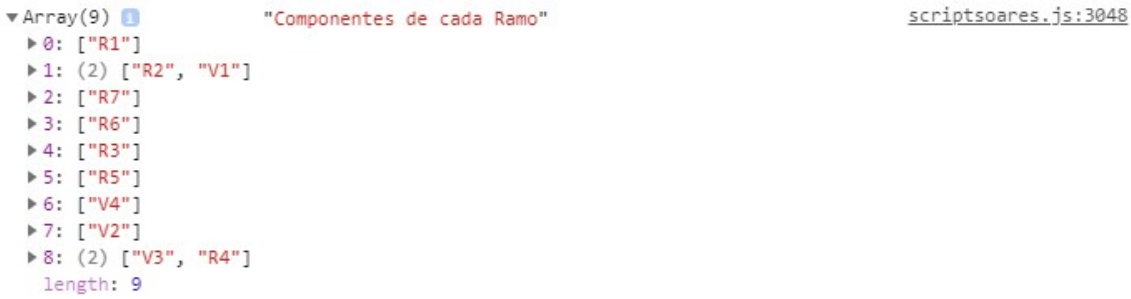


Figura 31 Vetores Componentes de cada Ramo

A Figura 32 retrata um exemplo do resultado da junção do que foi anteriormente explicado, em que o Ramo e a sua respetiva Corrente encontram-se numerados pelo mesmo índice numérico.

Ramo 1 constituído por: R2,V1 |-|-| I1: gnd → A

Figura 32 Exemplo do *layout* da identificação das Correntes nos Ramos

De seguida procede-se à geração das equações dos nós. Nesta fase, tem de se considerar quais são as associações feitas anteriormente pelo *script* entre os Nós reais e os Ramos do circuito. Em suma, é exigida uma verificação de quais os Ramos que estão ligados diretamente a cada Nó real. Depois de obtida esta informação, será necessário conhecer o sentido da corrente em cada ramo, arbitrado anteriormente, de forma a ser possível a atribuição de um sinal positivo ou negativo à corrente na respetiva equação do Nó. Por

definição, optou-se por construir as equações dos Nós pondo todas as correntes no primeiro membro da equação sendo que o segundo membro é constituído apenas por um 0. Assim sendo, tendo também em consideração a lógica usada anteriormente na definição do sentido das correntes nos Ramos, todas as correntes que correntes que convergem para o Nó serão antecedidas de um sinal positivo e todas as correntes que divergem deste serão antecedidas de um sinal negativo. Concluindo, para confirmar a convergência ou a divergência da corrente em relação ao Nó a ser analisado basta apenas verificar se o Nó em questão se encontra na primeira ou segunda posição do vetor nos_limitantes. Se se encontrar na primeira posição está-se perante uma divergência sendo atribuído um sinal negativo, se não, trata-se de uma convergência sendo atribuído um sinal positivo. A Figura 33 demonstra o resultado final da equação do Nó A, Nó constituinte do circuito da Figura 29.

Nó A:

$$-I0 - I3 + I1 + I2 = 0$$

Figura 33 Exemplo do *layout* das equações dos Nós

Por fim, resta definir as equações das Malhas. O processo de construção de equações das Malhas recai sobre um ciclo contínuo de análise que poderá ser apresentado sob a forma de um fluxograma da Figura 34.

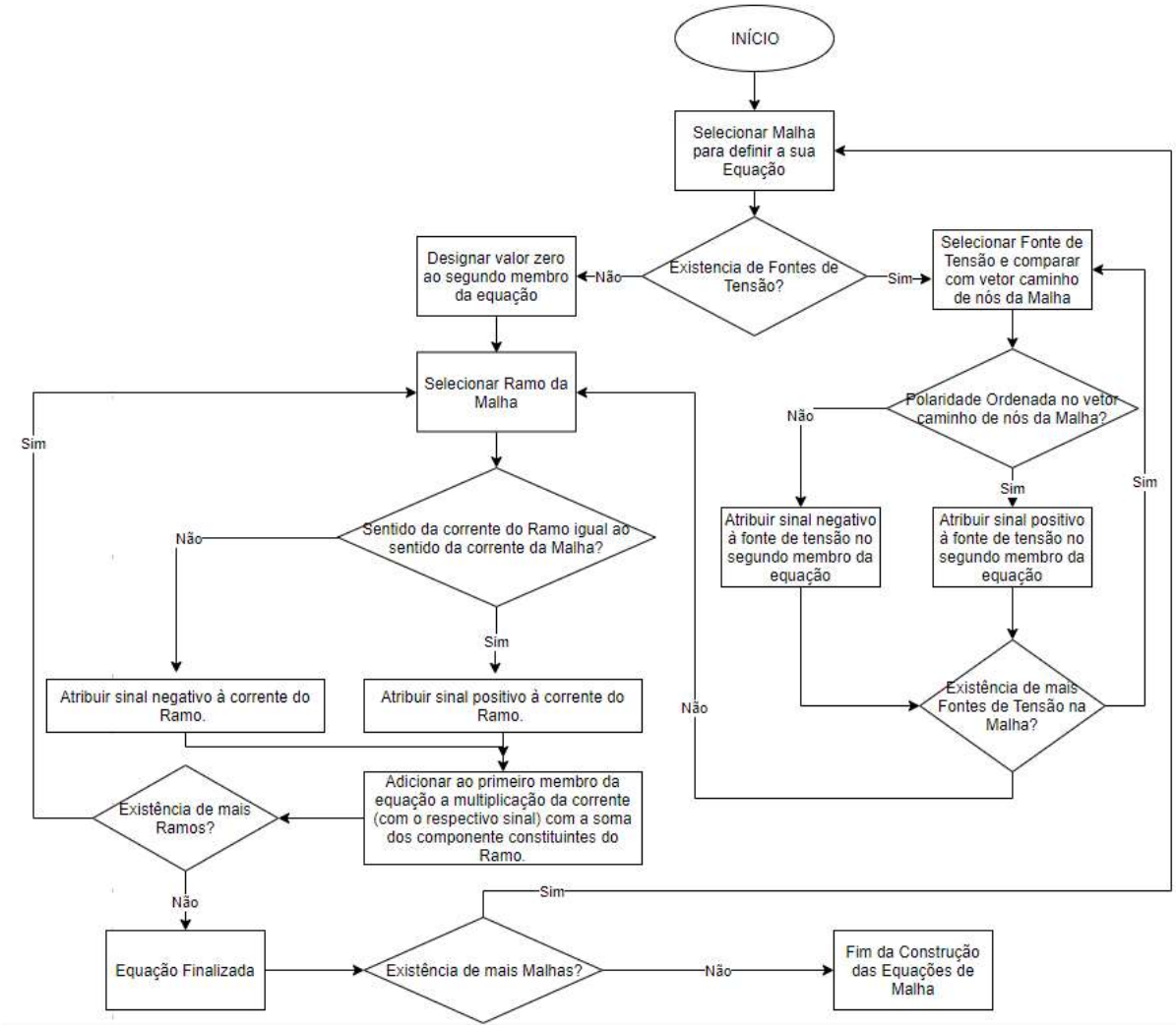


Figura 34 Fluxograma do algoritmo de identificação das Malhas

O funcionamento deste processo passa por comparar o sentido da corrente de Malha, que é arbitrado aleatoriamente pelo *script*, com o sentido da corrente de cada Ramo constituinte dessa mesma Malha. Neste processo, recorre-se ao percurso que cada Malha percorre verificando qual é a ordem dos Nós pelos quais a corrente de Malha passa. A Figura 35 apresenta um exemplo de um conjunto de Malhas selecionadas aleatoriamente pelo *script,* com base no circuito anterior, seguido do respetivo caminho de Nós que cada uma respeita.

Figura 35 Vetor conjunto de Malhas selecionadas e vetor Caminho de Nós percorrido por essas mesma Malhas

Para demonstrar, tome-se como exemplo o caso da Malha [4,5,7]. O *script* começa por comparar o primeiro Nó do vetor Nós limitantes do Ramo correspondente ao Ramo 4 com o primeiro Nó correspondente ao Ramo 4 no vetor Caminho de nós. A Figura 36 demonstra o explicado anteriormente.

Figura 36 Desigualdade entre Nós do vetor Caminho de Nós e Nós Limitantes do Ramo

Não se verifica uma igualdade, o que significa que o sentido da Corrente de Malha é contrário ao sentido da Corrente arbitrado nesse Ramo. Ou seja, o sinal atribuído à corrente desse Ramo na equação da Malha em questão irá ser negativo. Se a igualdade se verificasse o sinal seria positivo. De forma a facilitar a leitura e a reduzir ao máximo a equação optou-se por escrever sempre a Corrente, antecedida pelo seu respetivo sinal, seguida pela soma algébrica dos componentes constituintes desse Ramo, como demonstra a Figura 37. De

referir que todas as quedas de tensão são escritas no primeiro membro da equação. Este processo repete-se continuadamente para cada Ramo da Malha, sendo que a comparação é sempre feita entre o primeiro Nó do vetor Nós Limitantes do Ramo e o primeiro Nó do vetor Caminho de Nós correspondente ao Ramo a ser analisado.

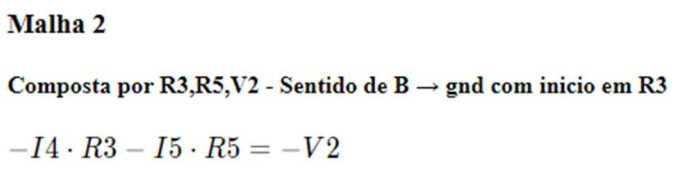


Figura 37 Exemplo do *layout* de uma equação de Malha

Posteriormente ao tratamento da primeira parte da equação, torna-se necessário definir o sentido das Fontes de Tensão para cada Malha. Para tal, compara-se o caminho da Corrente de Malha com a polaridade da Fonte no Ramo onde está contida. Por definição, o *QUCS* escreve sempre o polo positivo seguido do polo negativo na *netlist*. O facto de isto se verificar sempre que o simulador gera a *netlist* vai refletir na posição dos Nós armazenados na matriz de dados, isto é, o polo positivo será sempre armazenado na segunda posição e o polo negativo na terceira. Isto possibilita que o *script* possa analisar sempre da mesma forma o vetor dos nós associados às Fontes de Tensão. A mesma lógica aplicada na análise às quedas de Tensão que ocorrem nos Ramos é utilizada também neste processo. Contudo, esta lógica será traduzida na escrita sob a forma de Forças Eletromotrizes no segundo membro da equação.

Na hipótese de estarem incluídas mais do que uma Fonte de Tensão na mesma Malha, estão são reunidas numa *string* com o sinal de soma ou subtração, dependendo das suas orientações em relação à Malha que as engloba. No final, estas são armazenadas num vetor auxiliar.

Por fim, reúnem-se todos os dados recolhidos das operações descritas anteriormente, igualando numa *string* o primeiro membro da equação, referente às quedas de tensão, às Forças Eletromotrizes armazenadas no segundo membro da equação. O exemplo do resultado final da equação Malha {4,5,7} é representado na Figura 37.

4.4. RESOLUÇÃO DE SISTEMA DE EQUAÇÕES

Pese embora inicialmente estivesse apenas previsto a apresentação das equações, esta aplicação permite igualmente a resolução das equações de malha. Desta forma pretende-se que a mesma se revista de uma maior utilidade aos utilizadores.

Recorreu-se à biblioteca *Nerdamer* [18], atendendo a que a mesma é, neste momento, a que se apresenta como mais completa para *JavaScript*, permitindo desenvolver o processo de preparação das *strings* para posterior adaptação com o *solver*.

Na figura 39 são exemplificadas algumas operações com a biblioteca.

```
var sol = nerdamer.solveEquations('x^3+8=x^2+6','x');
console.log(sol.toString());
//1+i,-i+1,-1
```

```
var e = nerdamer.solveEquations('x^2+4-y', 'y');
console.log(e[0].text());
//4+x^2
```

```
var sol = nerdamer.solveEquations('x^2+8+y=x+6','x');
console.log(sol.toString());
//0.5*((-4*y-7)^0.5+1),0.5*(-(-4*y-7)^0.5+1)
```

```
var sol = nerdamer.solveEquations('cos(x)+cos(3*x)=1','x');
console.log(sol.toString());
//5.7981235959208695,0.4850617112587174
```

Figura 39 Exemplos de manipulação da biblioteca *Nerdamer*

Durante a definição das equações das Malhas, o algoritmo aproveita e copia todas as *strings* geradas, substituindo o nome dos componentes pelos seus valores constantes na matriz inicial. De seguida, trata-se de dividir a *string* do sistema de equações por múltiplas *strings* correspondentes a cada equação de Malha.

Uma ferramenta fulcral desta biblioteca foi o facto de esta permitir definir o número imaginário na notação comum em EE, aceitando desta forma as variáveis de Corrente de Malha "j_xx" onde "xx" corresponde ao ID da sua Malha. A Figura 40 apresenta um exemplo de uma operação da biblioteca *Nerdamer* com um número complexo *j*.

```
nerdamer.set('IMAGINARY', 'j');
x = nerdamer('sqrt(-1)');
console.log(x.toString());
```

Figura 40 Exemplo de operação com *Nerdamer*

Para realizar a resolução dos sistemas de equações, o *Nerdamer*, após a chamada da função "nerdamer.solveEquations()", retorna o vetor com os valores das Correntes das Malhas.

Refira-se que não obstante ter-se tentado a resolução de equações com números complexos, no caso da análise de circuitos em Corrente Alternada, tal não se mostrou possível, pelo que apenas é possível apresentar a resolução de equações quando se está perante uma análise de circuitos em Corrente Contínua.

4.5. APRESENTAÇÃO DE EQUAÇÕES EM LATEX

Para a apresentação de resultados, recorreu-se à biblioteca *KaTeX* [19], que permite adaptar o formato de escrita *LaTeX* [20], através de implementação *JavaScript*, para páginas HTML. Utilizou-se a funcionalidade "nerdamer.toTeX()" da ferramenta *Nerdamer*, funcionalidade essa que transforma qualquer *string* de formato de texto simples em formato *LaTeX*.

Na Figura 41, está representada uma comparação entre o *output* obtido em formato de texto e em configuração *LaTeX* [21].

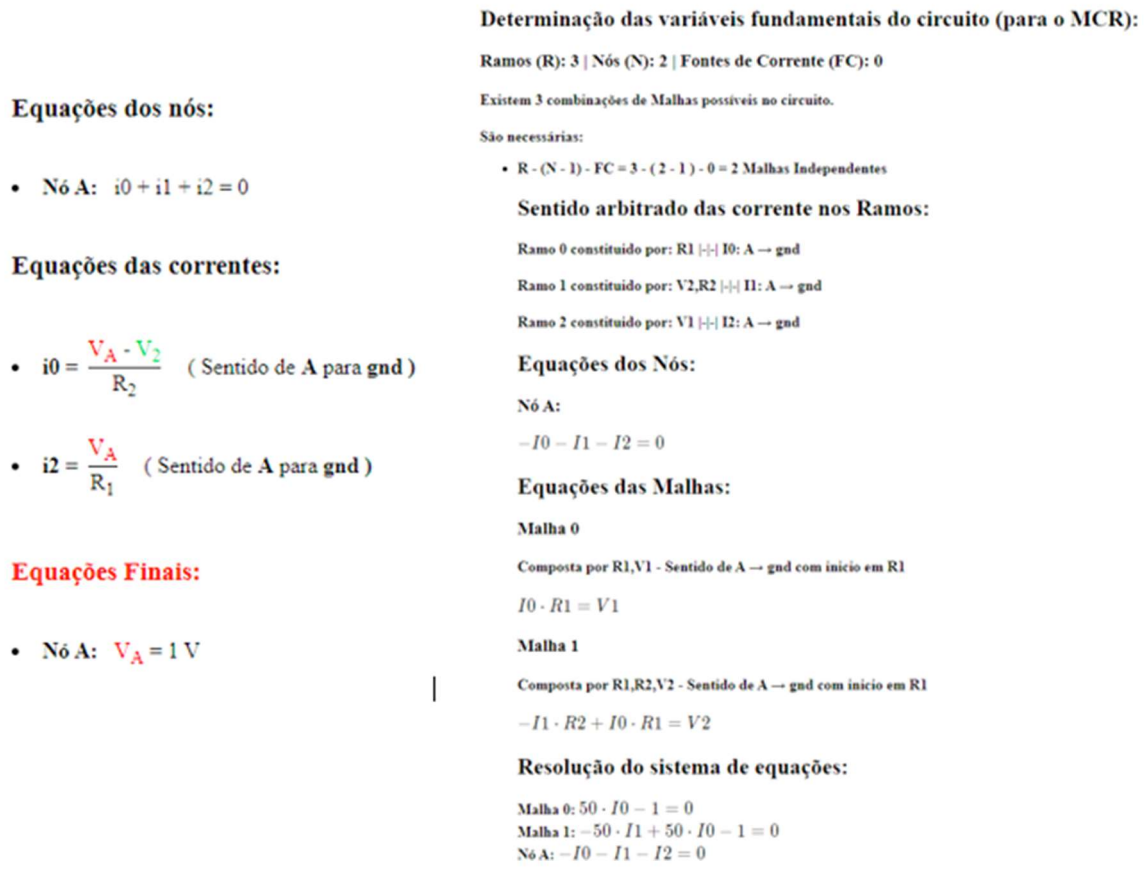


Figura 41 Comparação de *output* em formato de texto e LaTeX

A principal vantagem no uso desta ferramenta apoia-se nos factos de permitir uma renderização síncrona na página HTML de resultados, não necessitar de dependências (outras bibliotecas) e ser eficiente a processar *strings* formatadas em *LaTeX* para a devida apresentação gráfica.

5. EXEMPLOS DE RESULTADO DA APLICAÇÃO

Neste capítulo procede-se à exemplificação da aplicação, recorrendo a casos exemplo quer em Corrente Contínua, quer em corrente alternanda. Desta forma demonstra-se o correto funcionamento da aplicação. Também através dos exemplos (que serão integrados na aplicação), procura-se que os utilizadores vejam esta ferramenta como dotada de simplicidade, procurando dessa forma estimular a auto-aprendizagem.

Assim, todos os circuitos facultados são acompanhados do seu esquemático do QUCS, a *netlist* do circuito e uma imagem do circuito.

5.1. CASOS-EXEMPLO EM CORRENTE CONTÍNUA

Os circuitos que ora se apresentam estão ordenados por ordem de dificuldade, começando por circuitos simples e avançando para circuitos complexos.

Na Figura 42 podemos ver representado um circuito de nível básico, apresentando-se de seguida resultado obtido através da plataforma U=RISOLVE na Figura 43.

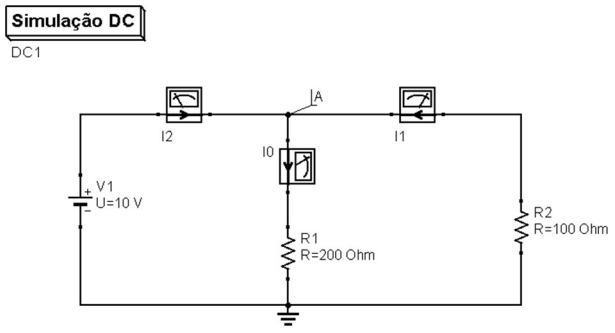


Figura 42 Circuito de nível básico em Corrente Contínua

Determinação das variáveis fundamentais do circuito (para o MCR):

Ramos (R): 3 | Nós (N): 2 | Fontes de Corrente (FC): 0

Existem 3 combinações de Malhas possíveis no circuito.

São necessárias:

- **R - (N - 1) - FC = 3 - (2 - 1) - 0 = 2 Malhas Independentes**

Sentido arbitrado das corrente nos Ramos:

Ramo 0 constituído por: R1 |-|-| I0: A → gnd

Ramo 1 constituído por: R2 |-|-| I1: gnd → A

Ramo 2 constituído por: V1 |-|-| I2: gnd → A

Equações dos Nós:

Nó A:

$$-I0 + I1 + I2 = 0$$

Equações das Malhas:

Malha 0

Composta por R1,R2 - Sentido de A → gnd com inicio em R1

$$I0 \cdot R1 + I1 \cdot R2 = 0$$

Malha 1

Composta por R1,V1 - Sentido de A → gnd com inicio em R1

$$I0 \cdot R1 = V1$$

Resolução do sistema de equações:

Malha 0: $100 \cdot I1 + 200 \cdot I0 = 0$
Malha 1: $200 \cdot I0 - 10 = 0$
Nó A: $-I0 + I1 + I2 = 0$

Resultados:

$I0$ = 0,050 A
$I1$ = - 0,100 A
$I2$ = 0,150 A

Figura 43 Solução MCR do circuito de nível básico em Corrente Contínua

Na Figura 44, encontra-se representado o segundo circuito em Corrente Continua, com nível de complexidade intermédio. A solução pela análise MCR para o respetivo circuito está demonstrada na Figura 45 e na Figura 46.

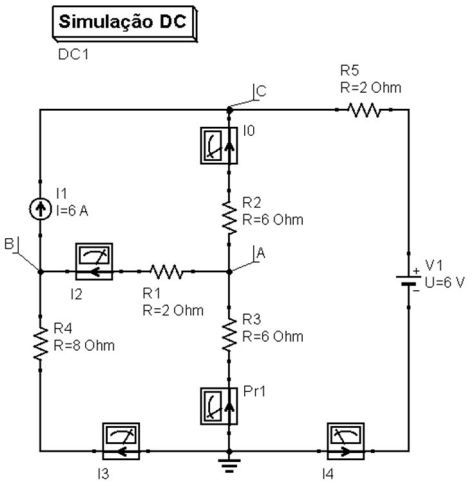


Figura 44 Circuito de nível intermédio em Corrente Contínua

Determinação das variáveis fundamentais do circuito (para o MCR):

Ramos (R): 6 | Nós (N): 4 | Fontes de Corrente (FC): 1

Existem 7 combinações de Malhas possíveis no circuito.

São necessárias:

- **R - (N - 1) - FC = 6 - (4 - 1) - 1 = 2 Malhas Independentes**

Sentido arbitrado das corrente nos Ramos:

Ramo 0 constituÂdo por: R2 |-|-| I0: A → C

Ramo 1 constituÂdo por: R3 |-|-| I1: gnd → A

Ramo 2 constituÂdo por: R1 |-|-| I2: A → B

Ramo 3 constituÂdo por: R4 |-|-| I3: gnd → B

Ramo 4 constituÂdo por: V1,R5 |-|-| I4: gnd → C

Equações dos Nós:

Nó C:

$$I0 + I4 + 6 = 0$$

Nó B:

$$I2 + I3 - 6 = 0$$

Nó A:

$$-I0 - I2 + I1 = 0$$

Figura 45 Solução MCR do circuito de nível intermédio em Corrente Contínua (parte1)

Equações das Malhas:

Malha 0

Composta por R2,R1,R4,V1,R5 - Sentido de C → A com inicio em R2

$$-I0 \cdot R2 - I3 \cdot R4 + I2 \cdot R1 + I4 \cdot R5 = V1$$

Malha 1

Composta por R2,R3,V1,R5 - Sentido de C → A com inicio em R2

$$-I0 \cdot R2 - I1 \cdot R3 + I4 \cdot R5 = V1$$

Resolução do sistema de equações:

Malha 0: $-6 \cdot I0 - 8 \cdot I3 + 2 \cdot I2 + 2 \cdot I4 - 6 = 0$
Malha 1: $-6 \cdot I0 - 6 \cdot I1 + 2 \cdot I4 - 6 = 0$
Nó C: $I0 + I4 + 6 = 0$
Nó B: $I2 + I3 - 6 = 0$
Nó A: $-I0 - I2 + I1 = 0$

Resultados:

$I0$ = - 3,064 A
$I1$ = 1,085 A
$I2$ = 4,149 A
$I3$ = 1,851 A
$I4$ = - 2,936 A

Figura 46 Solução MCR do circuito de nível intermédio em Corrente Contínua (parte2)

A Tabela 2 foi obtida a partir de amperímetros colocados em todos os Ramos do circuito em tem como objetivo validar os valores das Correntes nos Ramos produzidos pela aplicação.

Tabela 2 Validação dos resultados do circuito intermédio de análise em CC

number	I0.I	Pr1.I	I2.I	I3.I	I4.I
1	-3.06	1.09	4.15	1.85	-2.94

Para finalizar a demonstração da aplicação para circuitos em Corrente Contínua, apresenta-se o circuito da Figura 46, correspondente ao nível avançado, juntamente com a solução resultante da análise MCM representada na Figura 47.

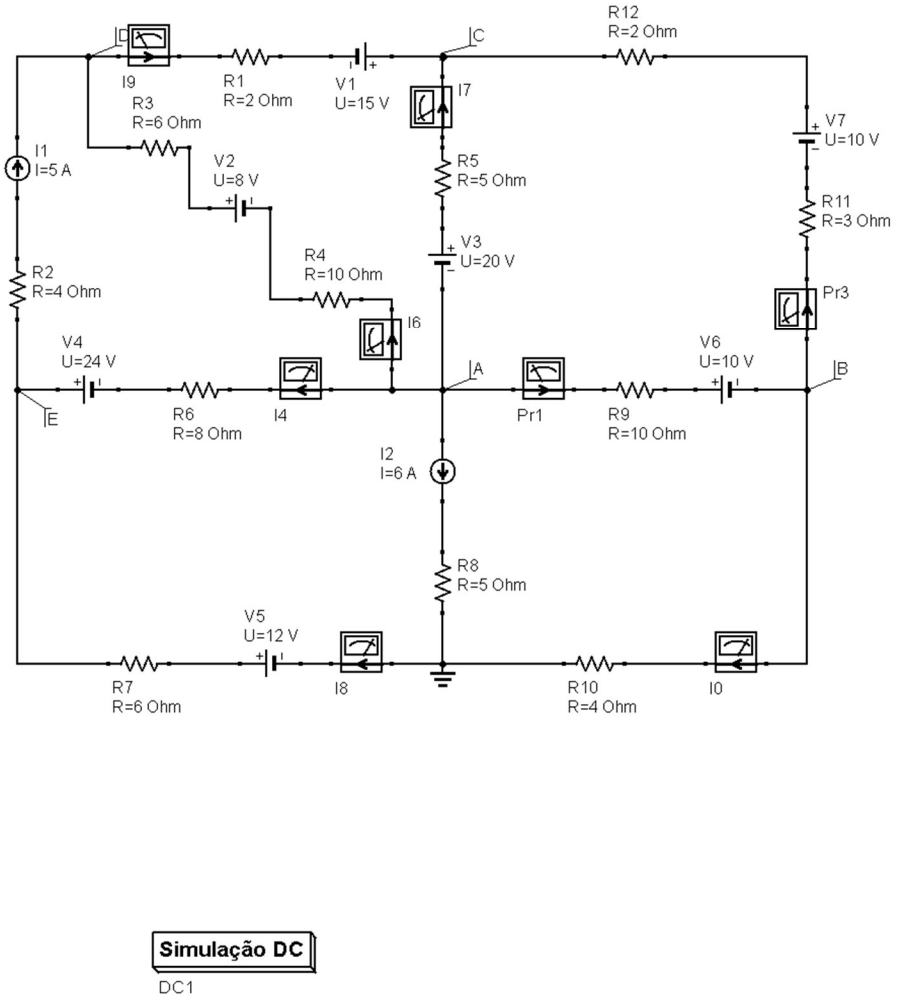


Figura 47 Circuito de nível avançado em Corrente Contínua

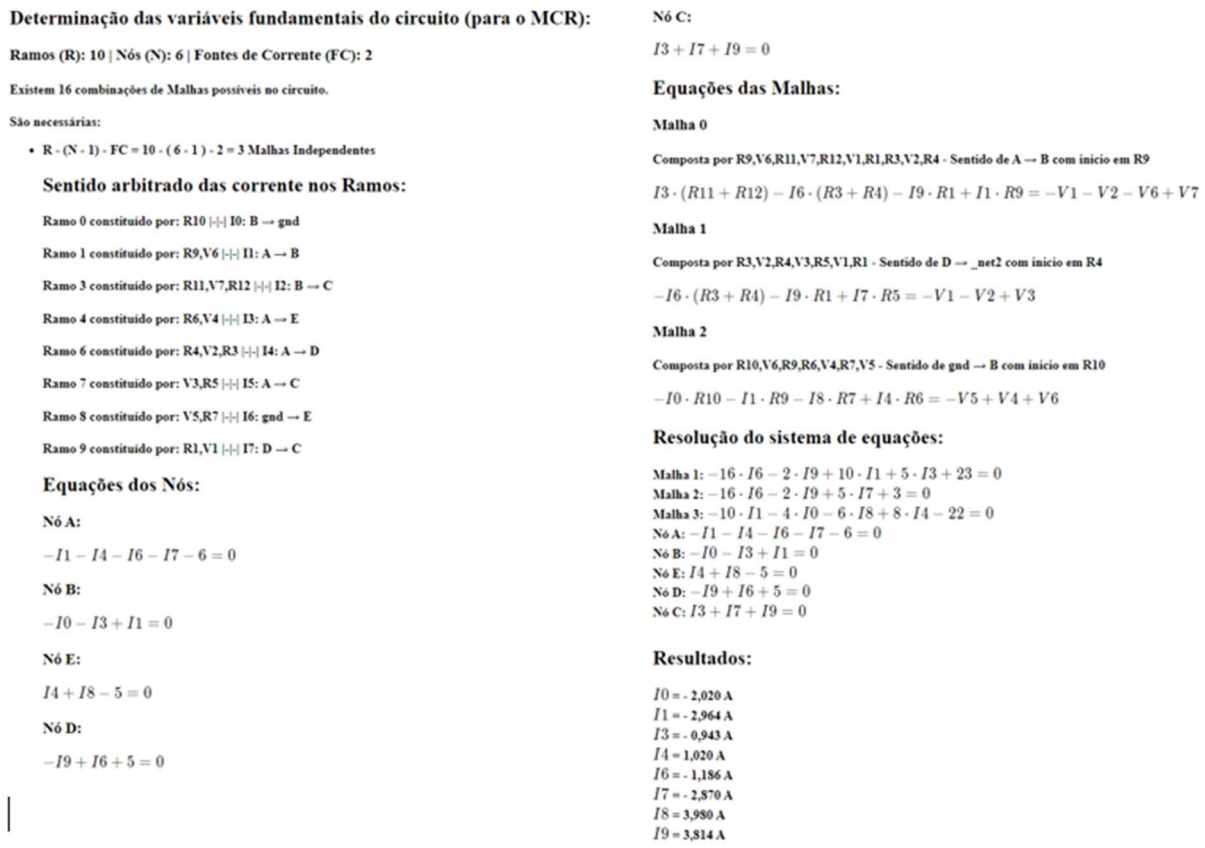


Figura 48 Solução MCR do circuito nível avançado em Corrente Contínua

A Tabela 3 abaixo foi obtida a partir de amperímetros colocados em todos os Ramos do circuito em tem como objetivo validar os valores das Correntes nos Ramos produzidos pela aplicação.

Tabela 3 Validação dos resultados do circuito avançado de análise em CC

number	I0.I	Pr1.I	Pr3.I	I4.I	I6.I	I7.I	I8.I	I9.I
1	-2.02	-2.96	-0.943	1.02	-1.19	-2.87	3.98	3.81

5.2. CASOS-EXEMPLO EM CORRENTE ALTERNADA

Agora em Corrente Alternada, os circuitos foram novamente dispostos por ordem de complexidade, lembrando que nestes, o processo de resolução das Correntes nos Ramos foi excluído. O primeiro circuito encontra-se representado na Figura 49 e o resultado obtido através da plataforma U=RISOLVE na Figura 50 e na Figura 51.

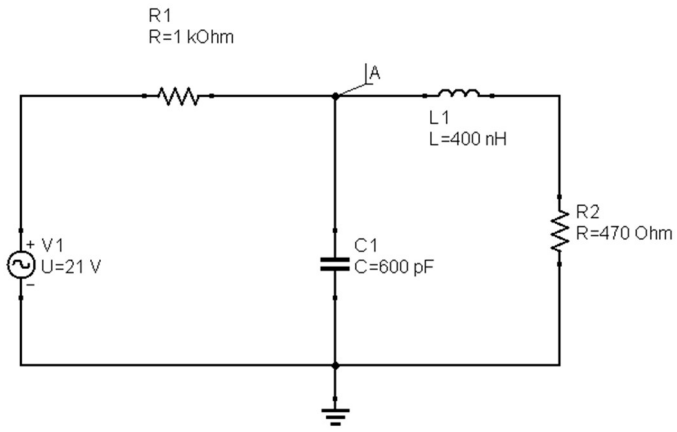


Figura 49 Circuito de nível básico em Corrente Alternada

Determinação das variáveis fundamentais do circuito (para o MCR):

Ramos (R): 3 | Nós (N): 2 | Fontes de Corrente (FC): 0

Existem 3 combinações de Malhas possíveis no circuito.

São necessárias:

- **R - (N - 1) - FC = 3 - (2 - 1) - 0 = 2 Malhas Independentes**

Sentido arbitrado das corrente nos Ramos:

Ramo 0 constituido por: C1 |-|-| I0: A → gnd

Ramo 1 constituido por: L1,R2 |-|-| I1: A → gnd

Ramo 2 constituido por: V1,R1 |-|-| I2: gnd → A

Figura 50 Solução MCR do circuito de nível básico em Corrente Alternada (parte1)

Equações dos Nós:

Nó A:

$$-I0 - I1 + I2 = 0$$

Equações das Malhas:

Malha 0

Composta por C1,V1,R1 - Sentido de A → gnd com inicio em C1

$$I0 \cdot \underline{Z}_{C1} + I2 \cdot R1 = V1$$

Malha 1

Composta por L1,R2,V1,R1 - Sentido de A → gnd com inicio em L1

$$I1 \cdot (\underline{Z}_{L1} + R2) + I2 \cdot R1 = V1$$

Figura 51 Solução MCR do circuito de nível básico em Corrente Alternada (parte1)

Na Figura 52, encontra-se representado o segundo circuito em Corrente Alternada, com nível de complexidade intermédio. A solução pela análise MCR para o respetivo circuito está demonstrada na Figura 53.

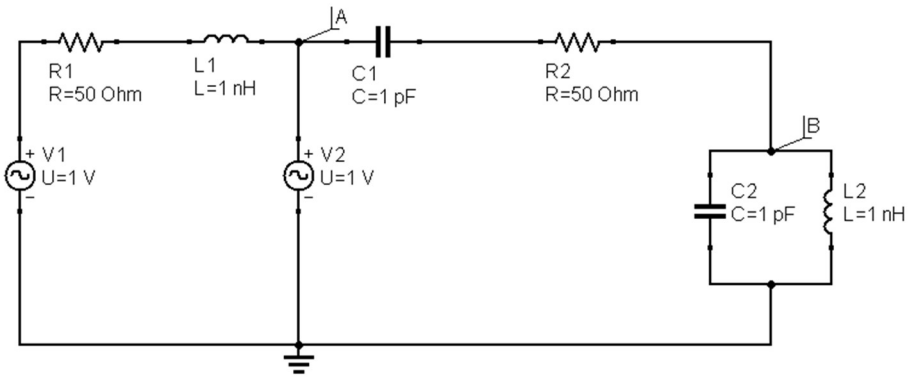


Figura 52 Circuito de nível intermédio em Corrente Alternada

Determinação das variáveis fundamentais do circuito (para o MCR):

Ramos (R): 5 | Nós (N): 3 | Fontes de Corrente (FC): 0

Existem 6 combinações de Malhas possíveis no circuito.

São necessárias:

- **R - (N - 1) - FC = 5 - (3 - 1) - 0 = 3 Malhas Independentes**

Sentido arbitrado das corrente nos Ramos:

Ramo 0 constituÃdo por: R2,C1 |-|-| I0: B → A

Ramo 1 constituÃdo por: C2 |-|-| I1: gnd → B

Ramo 2 constituÃdo por: L1,R1,V1 |-|-| I2: A → gnd

Ramo 3 constituÃdo por: V2 |-|-| I3: gnd → A

Ramo 4 constituÃdo por: L2 |-|-| I4: B → gnd

Equações dos Nós:

Nó A:

$$-I2 + I0 + I3 = 0$$

Nó B:

$$-I0 - I4 + I1 = 0$$

Equações das Malhas:

Malha 0

Composta por C1,R2,C2,V1,R1,L1 - Sentido de A → B com inicio em R2

$$-I0 \cdot (\underline{Z}_{C1} + R2) - I2 \cdot (\underline{Z}_{L1} + R1) - I1 \cdot \underline{Z}_{C2} = V1$$

Malha 1

Composta por C1,R2,C2,V2 - Sentido de A → B com inicio em R2

$$-I0 \cdot (\underline{Z}_{C1} + R2) - I1 \cdot \underline{Z}_{C2} = V2$$

Malha 2

Composta por R2,C1,L1,R1,V1,L2 - Sentido de B → A com inicio em R2

$$I0 \cdot (\underline{Z}_{C1} + R2) + I2 \cdot (\underline{Z}_{L1} + R1) - I4 \cdot \underline{Z}_{L2} = -V1$$

Figura 53 Solução MCR do circuito de nível intermédio em Corrente Alternada

Para concluir, apresenta-se o circuito da Figura 54, correspondente ao nível avançado para a análise em Corrente Alternada, juntamente com a solução resultante da análise MCR representada na Figura 55 e na Figura 56.

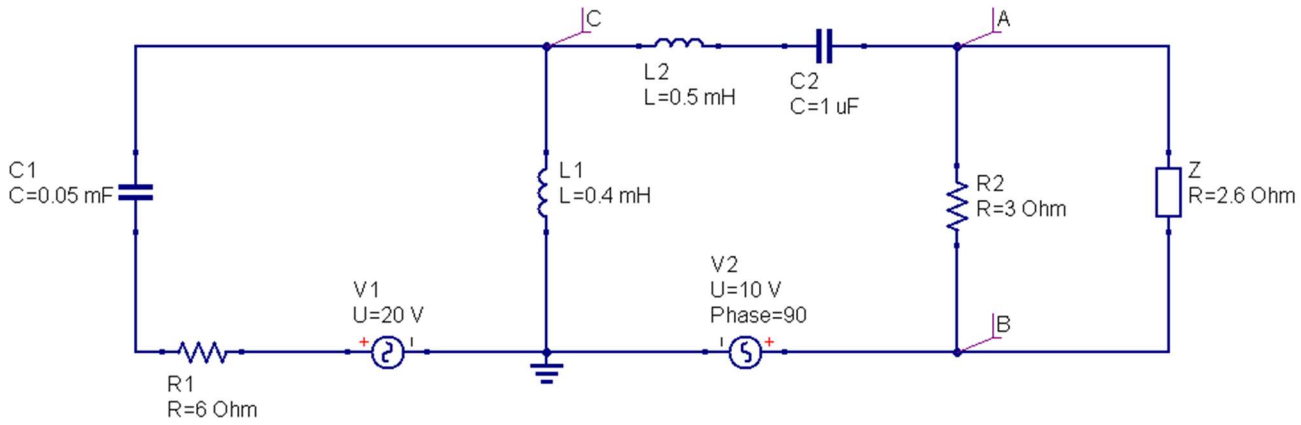


Figura 54 Circuito de nível avançado em Corrente Alternada

Equações dos Nós:

Nó A:

$$-I0 - I1 + I2 = 0$$

Equações das Malhas:

Malha 0

Composta por C1,V1,R1 - Sentido de A → gnd com inicio em C1

$$I0 \cdot \underline{Z}_{C1} + I2 \cdot R1 = V1$$

Malha 1

Composta por L1,R2,V1,R1 - Sentido de A → gnd com inicio em L1

$$I1 \cdot (\underline{Z}_{L1} + R2) + I2 \cdot R1 = V1$$

Figura 55 Solução MCR do circuito nível avançado em Corrente Alternada (parte 1)

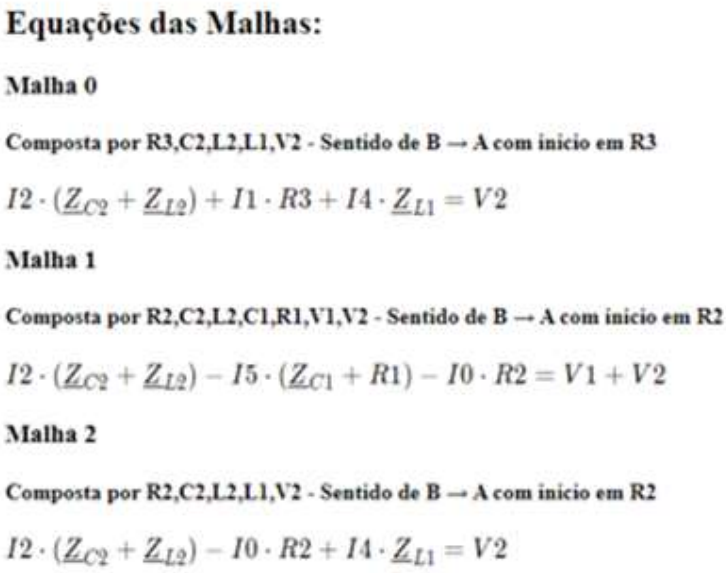


Figura 56 Solução MCR do circuito nível avançado em Corrente Alternada (parte 2)

A aplicação U=RISOLVE inclui este conjunto de circuitos exemplo, permitindo ao utilizador usufruir da ferramenta com maior facilidade, tendo disponível exemplos práticos para entender como deve proceder para produzir as suas próprias *netlists*.

Através dos mesmos exemplos de *netlists* facultadas, estes poderão ser testados tanto pelo MCR como pelo MTN e o MCR, valorizando assim o *software* desenvolvido atribuindo-lhe uma maior versatilidade.

6. CONCLUSÕES

Os alunos que dão os primeiros passos na EE deparam-se com uma dificuldade elevada na temática dos circuitos elétricos. Sendo esta uma disciplina chave, que serve de base a demais aprendizagens exigidas torna-se essencial que que os conceitos dessa disciplina sejam interiorizados e percebidos de forma correta. De forma a facilitar a aprendizagem de circuitos elétricos existem várias ferramentas de análise de circuitos elétricos existentes; no entanto estas não demonstram os passos necessários/equações matemáticas subjacentes ao bom funcionamento de um circuito elétrico, faltando-lhes a componente didática. Foi precisamente com este objetivo que se desenvolveu a presente aplicação.

Após análise das diversas ferramentas de simulação existentes no mercado, optou-se pelo QUCS como simulador base para a aplicação desenvolvida sobretudo porque este simulador é utilizado por muitos estudantes de Engenharia Eletrotécnica no ISEP. Assim, esta aplicação foi desenvolvida com o objetivo de dar continuidade e complementar o que outrora já foi desenvolvido pelos alunos da Licenciatura em Engenharia Eletrotécnica e Computadores [1][2][3][4][22], que pretende incluir novas funcionalidades ao *software* QUCS, para ajudar os alunos no seu percurso de aprendizagem e beneficiar o ensino. Assim sendo, a Aplicação para Geração da Metodologia/Equações do Método das Correntes nos Ramos a partir do QUCS, integra a plataforma U=RIsolve que contempla a Aplicação Metodologia/Equações do Método das Tensões nos Nós [1] e Aplicação para Geração da Metodologia/Equações do Método das Correntes nas Malhas [2], abrangendo também ferramentas de aquisição de dados ajustadas à versão alterada e melhorada do QUCS. Desta forma, o programa permite ler ficheiro de texto *netlist* gerados pela versão oficial como pela versão transformada do QUCS.

Com base no trabalho bibliográfico aprofundado relativo à produção do ficheiro de *netlist* por parte do QUCS, tornou-se exequível a melhoria da técnica de aquisição de dados, desenvolvendo novas formas de filtrar a informação necessária, como a obtenção de valores de componentes com prefixos do SI e a filtragem de resistências internas das Fontes de Tensão. Uma potencial melhoria ao projeto realizado passaria por uma adaptação dos processos de análise e aquisição de informações de *netlists* de outros simuladores existentes.

A criação de uma aplicação com capacidade de devolver ao utilizador as equações das Malhas obrigou a otimizar a informação adquirida na análise pelo método para poder transformar este conhecimento num algoritmo prático e exequível de implementar. O maior obstáculo passou por se descobrir qual a melhor abordagem à sua implementação. Após várias tentativas de elaborar um algoritmo com este propósito, surgiu a ideia de criar combinações de Ramos, de dois até ao número de Nós do circuito, de forma a produzir um vetor com todas as hipóteses de ligação entre Ramos. De seguida, este conjunto de combinações passam por um processo de validação e seleção automática.

A última fase de desenvolvimento resumiu-se na necessidade de prestar os resultados ao utilizador da forma mais clara e atrativa, com os resultados das equações propostas.

Para tal, utilizou-se as bibliotecas *Nerdamer* [18] e *KaTeX* [19], de forma a tornar possível a resolução dos sistemas de equações construídos pelo algoritmo e possibilitar a representação de todas as equações e operações em formato *LaTeX* [20], respetivamente,

Os objetivos deste projeto que foram propostos no começo foram alcançados, tendo em conta que, o *output* inicialmente previsto e delineado era apenas a apresentação das equações das do Método das Correntes nos Ramos. Entretanto, foi adicionado um novo objetivo: propôs-se que a aplicação produzisse os resultados da resolução dos sistemas de equações, isto é, o valor das Correntes nos Ramos. Este objetivo foi parcialmente alcançado dado que apenas se conseguiu obter os resultados para circuitos em Corrente Contínua pois não se encontrou, ainda, uma forma de resolver sistemas de equações de números complexos, que é necessário para a resolução de circuitos em Corrente Alternada.

O programa foi implementado com a finalidade de o partilhar na plataforma de hospedagem de código-fonte GitHub [23], de maneira a permitir o seu aproveitamento noutros projetos relacionados com o tema, permitindo outros desenvolvedores servirem-se de rotinas testadas de aquisição de dados a partir de ficheiros netlist, algoritmos depurados de análise e de exibição de resultados e, especialmente, do método de geração de Malhas e respetiva validação das mesmas.

Juntando ao facto de não existir nenhuma aplicação que defina as Malhas de um circuito e construa as suas equações automaticamente, este projeto mostra ser bastante recompensador do ponto de vista do desenvolvedor, por se refletir como uma ferramenta útil e única.

Durante a implementação do algoritmo, foi necessário estudar e investigar formas de contornar o problema da definição automática de Malhas a partir apenas de um ficheiro de texto. Nesta pesquisa, descobriu-se que o problema em questão figurava um problema definido na teoria da complexidade computacional designado NP-hard [24] (NP-difícil ou NP-complexo). Uma eventual otimização ao processo desenvolvido na geração de Malhas, passaria por aplicar soluções estudadas na teoria dos grafos, ao escrever um algoritmo Spanning Tree Protocol (STP), basando-se no algoritmo de busca em largura Breadth-First Search (BFS).

Concluindo, este projeto permitiu desenvolver diferentes apetências de natureza técnica, pela análise de circuitos, manipulação e estudo do simulador QUCS e programação *web*, estimulando um especial interesse pela linguagem JavaScript.

Referências Documentais

[1] SOUSA, Lino – *U=RISOLVE Aplicativo de Análise de Circuitos Elétricos Simulados no QUCS*. Dissertação de Licenciatura em Engenharia Eletrotécnica e Computadores orientada pelo Eng.º Mário Alves e Francisco Pereira e apresentada no Instituto Superior de Engenharia do Instituto Politécnico do Porto, em 2018.

[2] DUARTE, Miguel – *Aplicação para Geração da Metodologia/Equações do Método das Correntes nas Malhas, a partir do QUCS*. Dissertação de Licenciatura em Engenharia Eletrotécnica e Computadores orientada pelo Eng.º Mário Alves e Francisco Pereira e apresentada no Instituto Superior de Engenharia do Instituto Politécnico do Porto, em 2019.

[3] MARTINS, Nelson – *Desenvolvimento de Módulos para o Qucs*. Dissertação de Licenciatura em Engenharia Eletrotécnica e Computadores orientada pelo Eng.º Mário Alves e apresentada no Instituto Superior de Engenharia do Instituto Politécnico do Porto, em 2016.

[4] SILVA, Alberto – *Design, Implementation and Integration of new Functionalities in the QUCS Simulator.* Dissertação de Licenciatura em Engenharia Eletrotécnica e Computadores orientada pelos Eng. Mário Alves e Francisco Pereira e apresentada no Instituto Superior de Engenharia do Instituto Politécnico do Porto, em 2017.

[5] CIRCUITLAB – CircuitLab; https://www.circuitlab.com/

[6] EASYEDA – EasyEDA; https:// www.easyeda.com/

[7] EVERYCIRCUIT – EveryCircuit; http:// www.everycircuit.com/

[8] GITHUB – KTechLab; https:// www.github.com/ktechlab/ktechlab/

[9] ANALOG DEVICES – LTspice; https://www.analog.com/en/design-center/design-tools-and-calculators/ltspicesimulator.html

[10] GITHUB – Oregano; https://www.github.com/drahnr/oregano/

[11] PAUL FALSTAD – Falstad Circuit Simulator, https://www.falstad.com/circuit/

[12] POWERSIM – PSIM, https://powersimtech.com/products/psim/

[13] CADENCE – PSpice, http://www.pspice.com/

[14] SOURCEFORGE – QUCS, http://qucs.sourceforge.net/

[15] VIANA, Ana; PEREIRA, Francisco; ALVES, Mário – FEELE, Métodos de Análise e Simplificação de Circuitos Elétricos (Parte 1 e Parte 2), 2015.

[16] SOURCEFORGE – QUCS, *A Tutorial, Qucs Simulation of SPICE Netlists*; http://qucs.sourceforge.net/docs/tutorial/spicetoqucs.pdf/

[17] ROSETTACODE – Permutations with repetitions, www.rosettacode.org/

[18] NERDAMER – Nerdamer; https://www.nerdamer.com/

[19] KATEX – KaTeX; https://www.katex.org/

[20] LATEX – LaTeX; https://www.latex-project.org/

[21] KATEX – API KaTeX; https://www.katex.org/docs/api.html

[22] MACEDO, Pedro – *Análise Comparativa e Melhoramento de Simuladores de Circuitos Elétricos*. Dissertação de Licenciatura em Engenharia Eletrotécnica e Computadores orientada pelo Eng.º Mário Alves e apresentada no Instituto Superior de Engenharia do Instituto Politécnico do Porto, em 2015.

[23] GITHUB – GitHub; https://www.github.com/

[24] HOCHBAUM, Dorit - *Approximation Algorithms for NP-Hard Problems*, 1ª Edição, pp. 8-10, 1996