



**UFOP**

Universidade Federal  
de Ouro Preto

**Universidade Federal de Ouro Preto  
Instituto de Ciências Exatas e Aplicadas  
Departamento de Computação e Sistemas**

## **AccessCrawler - *Web Crawler* para Avaliar a Acessibilidade *Web***

**Filipe Nunes Soares**

### **TRABALHO DE CONCLUSÃO DE CURSO**

ORIENTAÇÃO:  
Dra. Lucinéia Souza Maia

**Agosto, 2023  
João Monlevade—MG**

**Filipe Nunes Soares**

**AccessCrawler - *Web Crawler* para Avaliar a  
Acessibilidade *Web***

Orientador: Dra. Lucinéia Souza Maia

Monografia apresentada ao curso de Sistemas de Informação do Instituto de Ciências Exatas e Aplicadas, da Universidade Federal de Ouro Preto, como requisito parcial para aprovação na Disciplina “Trabalho de Conclusão de Curso II”.

**Universidade Federal de Ouro Preto**

**João Monlevade**

**Agosto de 2023**

*Este trabalho é dedicado aos meus queridos pais e irmão*

# Agradecimentos

Gostaria de expressar minha sincera gratidão a todas as pessoas que desempenharam um papel significativo ao longo da minha jornada acadêmica. O apoio e o incentivo de cada um de vocês foram fundamentais para meu crescimento pessoal e profissional, e por isso, minha profunda apreciação.

A meu pai devo um agradecimento especial. Sua dedicação incansável e comprometimento desde minha infância até o presente momento foram essenciais para minha formação. Seus constantes sacrifícios moldaram minha determinação e me inspiraram a superar desafios.

Minha mãe, sua amorosa paciência e apoio inabalável me guiaram em todos os momentos. Mesmo quando eu estava longe de ser perfeito, você esteve ao meu lado, oferecendo seu carinho e cuidado. Sua força e resiliência durante o início da minha jornada universitária foram um farol de esperança, e sou imensamente grato por sua presença constante.

A meu irmão, agradeço por ser uma inspiração contínua em minha vida. Minha busca por excelência acadêmica foi motivada em grande parte pelo desejo de ser um modelo para você. Sua amizade e confiança em mim significam mais do que palavras podem expressar. Espero que você saiba que nossa ligação é uma das minhas maiores bênçãos.

Lucinéia, sua orientação durante o desenvolvimento do meu TCC foi inestimável. Sua paciência, conhecimento e disponibilidade para me ajudar em cada etapa foram cruciais para o sucesso deste trabalho. Seu papel como mentora não apenas me ajudou a superar desafios, mas também me ensinou lições valiosas que levarei comigo ao longo da vida.

As minhas amigas Thais e Ana Júlia, a amizade de vocês tem sido um raio de sol em minha jornada universitária. A presença nos bons momentos e nos difíceis trouxe conforto e alegria à minha vida. Sua disposição em me ajudar e ser uma fonte constante de apoio demonstra o verdadeiro significado da amizade, e por isso, estendo meu profundo agradecimento a vocês.

Por fim, à minha república Granja. Vocês tornaram-se minha segunda família durante esses cinco anos. O senso de comunidade, camaradagem e apoio mútuo que compartilhamos fez de João Monlevade um lugar mais acolhedor e especial. Agradeço a todos com quem tive o prazer de conviver, pelos momentos compartilhados e por tornarem minha experiência universitária memorável.

*“Só porque um homem não pode usar os olhos, não significa que ele não tem visão.”*

— Stevie Wonder

# Resumo

Este trabalho propõe o desenvolvimento e análise de um *Web Crawler* e um Avaliador de Acessibilidade, para avaliar a acessibilidade dentro do contexto do domínio da Universidade Federal de Ouro Preto. A acessibilidade *Web* é uma abordagem que busca assegurar que todos os usuários, incluindo aqueles com deficiência, possam acessar e interagir com a internet de forma inclusiva. Embora a plataforma *online* do Avaliador e Simulador de Acessibilidade em Sítios seja um instrumento acessível, sua limitação em avaliar apenas a página indexada restringe a análise completa da acessibilidade do domínio, o que suscita a indagação sobre a viabilidade de uma estratégia abrangente para avaliar todas as páginas de um domínio. No âmbito deste estudo, o principal objetivo é a concepção de um *Web Crawler* dedicado à avaliação de acessibilidade *Web* em todas as páginas do domínio da universidade. A abordagem pragmática da *Design Science Research* constitui a base metodológica deste trabalho, visando solucionar problemas práticos do mundo real por meio do desenvolvimento de artefatos. Os resultados obtidos a partir da execução do *Web Crawler* revelam um processo de coleta de *links* em múltiplos ciclos. No decorrer dos ciclos, o *Crawler* aumentou significativamente a quantidade de *links* coletados. Entretanto, à medida que os ciclos avançaram, observou-se uma desaceleração na taxa de aumento de *links* coletados, sugerindo uma possível saturação do espaço de busca disponível. A avaliação feita pelo ASES nos 7.607 *links* que obtiveram sucesso, revelou que apenas cerca de 0,54% atingiram o alto nível de conformidade desejado.

**Palavras-chave:** *Web Crawler*, Avaliador de Acessibilidade, acessibilidade *Web*.

# Abstract

This work proposes the development and analysis of a Web Crawler and an Accessibility Evaluator, aimed at assessing accessibility within the context of the domain of the Federal University of Ouro Preto. Web accessibility is an approach that seeks to ensure that all users, including those with disabilities, can access and interact with the internet in an inclusive manner. While the online platform of the Accessibility Evaluator and Simulator is an accessible tool, its limitation in evaluating only the indexed page restricts the comprehensive analysis of domain accessibility, raising the question of the viability of a comprehensive strategy to assess all pages within a domain. In the scope of this study, the primary objective is the design of a Web Crawler dedicated to evaluating web accessibility across all pages of the university's domain. The pragmatic approach of Design Science Research constitutes the methodological foundation of this work, aiming to solve real-world practical problems through the development of artifacts. The results obtained from the execution of the Web Crawler reveal a process of collecting links in multiple cycles. Over the course of these cycles, the Crawler significantly increased the quantity of collected links. However, as the cycles progressed, a slowdown in the rate of increase of collected links was observed, suggesting a potential saturation of the available search space. The evaluation conducted by ASES on the 7,607 links that succeeded revealed that only about 0.54% achieved the desired high level of compliance.

**Keywords:** Web Crawler, Accessibility Evaluator, Web accessibility.

# Lista de ilustrações

Figura 1 – Quadro DSR . . . . .	17
Figura 2 – Seções do e-MAG 3.1 . . . . .	20
Figura 3 – Relatório de conformidade da página inicial da UFOP de acordo com o e-MAG . . . . .	21
Figura 4 – Resultado da avaliação em formato PDF . . . . .	22
Figura 5 – Listagem dos Erros Mais Comuns, com destaque para a seção de erros do tipo Marcação . . . . .	25
Figura 6 – Avaliação dos <i>links</i> no arquivo <code>avaliacao.txt</code> . . . . .	29
Figura 7 – Porcentagem de avaliação . . . . .	38
Figura 8 – Tabela de Avisos e Erros . . . . .	38
Figura 9 – Desempenho de cada ciclo de processamento . . . . .	43
Figura 10 – Níveis de conformidade dos <i>links</i> avaliados . . . . .	44



# Lista de Códigos

4.1	Estrutura de diretórios e arquivos utilizada no projeto. . . . .	28
4.2	Método principal do programa. . . . .	30
4.3	Inicialização das <i>threads</i> e retorno dos <i>links</i> encontrados. . . . .	31
4.4	Processamento dos <i>links</i> . . . . .	32
4.5	Método encontrar <i>links</i> . . . . .	32
4.6	Gerar <i>links</i> absoluto. . . . .	33
4.7	Filtros de validação de <i>links</i> . . . . .	34
4.8	Processo de avaliação. . . . .	35
4.9	Classe Avaliação. . . . .	36
4.10	Método de inicialização dos navegadores. . . . .	37
4.11	Processo de avaliação. . . . .	37

# Lista de ilustrações

Figura 1 – Quadro DSR . . . . .	17
Figura 2 – Seções do e-MAG 3.1 . . . . .	20
Figura 3 – Relatório de conformidade da página inicial da UFOP de acordo com o e-MAG . . . . .	21
Figura 4 – Resultado da avaliação em formato PDF . . . . .	22
Figura 5 – Listagem dos Erros Mais Comuns, com destaque para a seção de erros do tipo Marcação . . . . .	25
Figura 6 – Avaliação dos <i>links</i> no arquivo <code>avaliacao.txt</code> . . . . .	29
Figura 7 – Porcentagem de avaliação . . . . .	38
Figura 8 – Tabela de Avisos e Erros . . . . .	38
Figura 9 – Desempenho de cada ciclo de processamento . . . . .	43
Figura 10 – Níveis de conformidade dos <i>links</i> avaliados . . . . .	44

# Lista de tabelas

Tabela 1 – Análise do processamento dos <i>links</i> pelo AccessCrawler . . . . .	42
Tabela 2 – Média de Erros e Avisos . . . . .	45

# Lista de abreviaturas e siglas

**UFOP** Universidade Federal de Ouro Preto

**W3C** *World Wide Web Consortium*

**e-MAG** Modelo de Acessibilidade em Governo Eletrônico

**SGD** Secretaria de Governo Digital

**ASES** Avaliador e Simulador de Acessibilidade em Sítios

**DSR** *Design Science Research*

**WAI** *Web Accessibility Initiative*

**WCAG** *Web Content Accessibility Guidelines*

**ABNT** Associação Brasileira de Normas Técnicas

**HTML** *HyperText Markup Language*

**PDF** *Portable Document Format*

**WWW** *World Wide Web*

**URL** *Uniform Resource Locator*

**HTTP** *HyperText Transfer Protocol*

**HTTPS** *HyperText Transfer Protocol Secure*

**EIAO** *European Internet Accessibility Observatory*

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>14</b>
<b>1.1</b>	<b>Justificativa/problema</b>	<b>14</b>
<b>1.2</b>	<b>Objetivo</b>	<b>15</b>
<b>1.3</b>	<b>Metodologia</b>	<b>15</b>
<b>1.4</b>	<b>Organização do Trabalho</b>	<b>17</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>18</b>
<b>2.1</b>	<b>Acessibilidade Web</b>	<b>18</b>
2.1.1	Recomendações de acessibilidade Web	18
2.1.2	Avaliação de acessibilidade Web utilizando a ferramenta ASES	20
<b>2.2</b>	<b>Web Crawler</b>	<b>22</b>
<b>3</b>	<b>TRABALHOS RELACIONADOS</b>	<b>24</b>
<b>3.1</b>	<b>Observatório da Acessibilidade da Web Brasileira</b>	<b>24</b>
<b>3.2</b>	<b><i>Towards dDistribution of Web Sites in a Crawler Used for Large-Scale Web Accessibility Assessment</i></b>	<b>25</b>
<b>3.3</b>	<b><i>Monitoring Accessibility of Governmental Websites in Europe</i></b>	<b>26</b>
<b>4</b>	<b>DESENVOLVIMENTO</b>	<b>27</b>
<b>4.1</b>	<b>Estrutura do Código</b>	<b>29</b>
4.1.1	Main.py	29
4.1.2	Arquivos.py	30
4.1.3	Processar_links.py	31
4.1.4	Encontrar_links.py	32
4.1.5	Filtro.py	34
4.1.6	Avaliador.py	35
<b>5</b>	<b>RESULTADOS</b>	<b>40</b>
<b>5.1</b>	<b>Web Crawler</b>	<b>40</b>
5.1.1	Análise do rastreo do AccessCrawler	41
<b>5.2</b>	<b>Análise dos resultados das avaliações de acessibilidade</b>	<b>43</b>
<b>6</b>	<b>CONCLUSÃO</b>	<b>47</b>
	<b>REFERÊNCIAS</b>	<b>48</b>

# 1 Introdução

A acessibilidade *Web* assume um papel crucial ao estabelecer práticas de desenvolvimento que visam facilitar a interação de todos os usuários com a *internet*, garantindo inclusão mesmo para aqueles com deficiência, ao proporcionar independência no uso da tecnologia. Conforme o *World Wide Web Consortium (W3C)*:

Acessibilidade na *Web* significa que pessoas com deficiência podem usar a *Web*. Mais especificamente, a acessibilidade na *Web* significa que pessoas com deficiência podem perceber, entender, navegar, interagir e contribuir para a *Web* (W3C, s.d.).

Para se adequar a esses princípios inclusivos, foram estabelecidas diretrizes. Uma delas é o Modelo de Acessibilidade em Governo Eletrônico (e-MAG), um conjunto de recomendações de acessibilidade *Web* desenvolvido pela Secretaria de Governo Digital (SGD) em 2005, como resposta ao Decreto-Lei nº 5.296. Tal decreto regula as Leis nº 10.048/2000 e nº 10.098/2000, as quais tratam da acessibilidade de pessoas com deficiência no âmbito do Governo Federal, estabelecendo critérios e normas para promover a acessibilidade e a inclusão dessas pessoas em diversos aspectos sociais (Brasil, 2004).

Para avaliar os *sites* governamentais em conformidade com o e-MAG, o Avaliador e Simulador de Acessibilidade em Sítios (ASES) foi desenvolvido. Inicialmente lançado em 2007 como uma versão *desktop* e, posteriormente, em 2016, em formato *Web*, o ASES direciona sua atenção para avaliar as diretrizes do e-MAG 3.1. Essa iniciativa é de notável relevância ao incentivar a acessibilidade *Web*, permitindo que empresas e órgãos governamentais desenvolvam produtos e serviços acessíveis para um público mais vasto, abrangendo também as pessoas com deficiência (PLANEJAMENTO et al., 2016).

No entanto, mesmo com as ferramentas e tecnologias disponíveis para auxiliar programadores a desenvolver projetos acessíveis, muitos ainda não adotam esses recursos por falta de conhecimento dos programadores e publicadores de conteúdo como sendo os maiores empecilhos (SILVA; RODRIGUES, 2018).

## 1.1 Justificativa/problema

A ferramenta ASES é disponibilizada em uma plataforma *online* (Governo Federal do Brasil, 2016), a qual é possível que um usuário insira um *link*, faça o *upload* de um arquivo ou código fonte a ser avaliado identificando os erros e avisos de acessibilidade com base nos critérios de avaliação estabelecidos pela ferramenta a partir do e-MAG. Além disso, o ASES mostra a porcentagem em conformidade de acordo com a recomendação supracitada.

Vale ressaltar que a ferramenta avalia apenas a página indexada, o que impede a verificação da experiência real de um usuário com deficiência em relação à acessibilidade do *site* como um todo. Ao realizar a análise de todas as páginas de um subdomínio, é possível detectar problemas significativos de acessibilidade *Web* que não foram identificados na verificação inicial do *link*. Dessa forma, é essencial verificar todas as páginas para obter uma apuração mais precisa e completa dos resultados em relação à acessibilidade do *site*.

Há casos em que é difícil avaliar todos os *sites* de um domínio, especialmente quando a instituição não tem controle total sobre todos os subdomínios e *links* externos associados às páginas registradas, como é o caso da Universidade Federal de Ouro Preto (UFOP). Nesse sentido a seguinte questão de problema foi colocado: é possível criar uma estratégia para avaliar todas as páginas de um domínio?

## 1.2 Objetivo

O objetivo deste trabalho consiste em desenvolver um *Web Crawler* para avaliar a acessibilidade *Web* de todas as páginas do domínio da UFOP. Os objetivos específicos são:

- Mapear e rastrear os *links* que constam no domínio da UFOP.
- Avaliar a acessibilidade de cada página do domínio.
- Compilar os resultados apresentados pelo ASES.

## 1.3 Metodologia

Este trabalho utiliza o *Design Science Research* (DSR) como base metodológica. A abordagem pragmática da DSR busca identificar e compreender problemas do mundo real, propondo soluções apropriadas que avancem o conhecimento teórico da área, sem buscar alcançar verdades últimas ou grandes teorias (BAX, 2013).

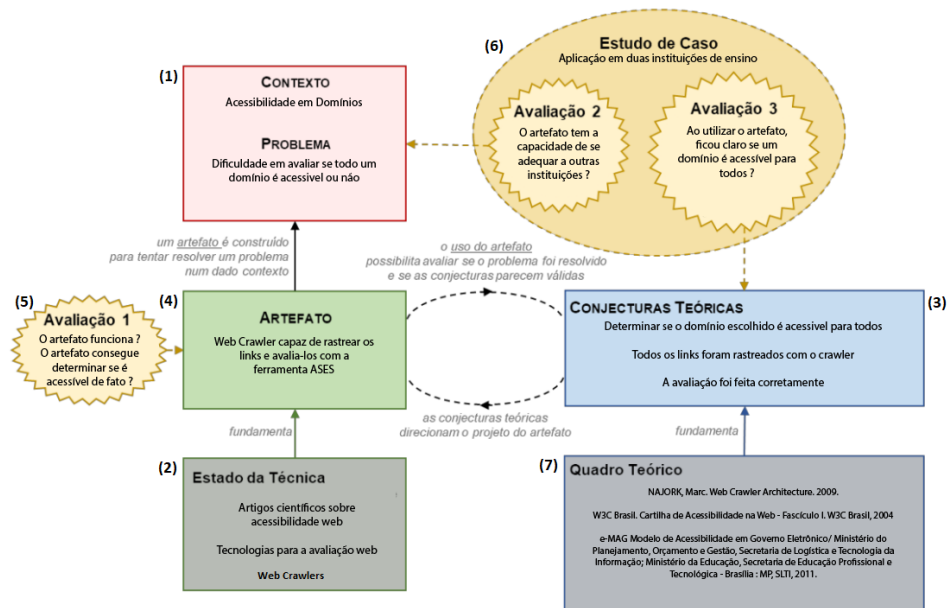
O pesquisador na DSR tem dois objetivos fundamentais: resolver um problema prático em um contexto específico por meio do desenvolvimento de um artefato e gerar novo conhecimento científico. Nesse caso o artefato será o *Web Crawler* chamado: AccessCrawler.

Esses objetivos são alcançados por meio de dois ciclos de pesquisa inter-relacionados. O Ciclo de Engenharia (*Design*) concentra-se no projeto e desenvolvimento de um artefato para solucionar o problema, realizando avaliações para identificar melhorias e aprimorar o projeto. Já o Ciclo Empírico (do Rigor) baseia-se em teorias e métodos científicos para garantir que a pesquisa seja conduzida de acordo com os padrões de rigor teórico e metodológico exigidos em uma investigação científica (PIMENTEL; FILIPPO; SANTORO, 2019).



A Figura 1 mostra uma representação DSR de Pimentel adaptada para este TCC. É uma representação visual utilizada para facilitar a organização e comunicação dos diversos componentes do projeto. Os elementos principais constituem: (1) Contexto e Problema, onde são identificados o contexto em que o problema ocorre e a questão que precisa ser resolvida. No caso desse TCC, o contexto é acessibilidade em domínios e o problema é a dificuldade em avaliar se todo um domínio é acessível ou não; (2) estado da Técnica, é realizada uma revisão da literatura e do conhecimento existente sobre o problema em questão. O objetivo é entender o que já foi feito, quais abordagens foram tentadas e quais lacunas existem na solução do problema. Nesse caso foram pesquisados artigos científicos tecnologias para avaliação de acessibilidade e Web Crawlers; (3) quadro Teórico, apresenta a evolução das conjecturas teóricas, refletindo sobre como a solução proposta se relaciona com a literatura existente e como ela contribui para o avanço do conhecimento; Nesse TCC, o quadro teórico são as publicações (PARDINI et al., 2021; CHEN; WANG; OLSEN, ; BÜHLER et al., 2008); (4) conjecturas Teóricas, são formuladas as hipóteses ou conjecturas teóricas que embasam o desenvolvimento do artefato. Elas servem como diretrizes teóricas para a criação da solução, um Web Crawler desenvolvido em Python integrado com o ASES é capaz de avaliar a acessibilidade de um domínio Web; (5) artefato, é o núcleo da pesquisa. O artefato é a solução prática que está sendo desenvolvida para resolver o problema identificado. Pode ser um *software*, um processo, um método, uma ferramenta, entre outros. Nesse caso, o artefato é o AccessCrawler; (6) avaliação 1, o artefato é avaliado quanto à sua viabilidade, eficácia e adequação à solução do problema. Pode envolver testes, experimentos ou outros métodos de avaliação. Nesse caso a avaliação determinará é o AccessCrawler consegue rastrear todo o domínio e se ele consegue avaliar a acessibilidade das páginas rastreadas; (7) estudo de Caso, se apropriado um estudo de caso pode ser conduzido para avaliar o artefato em um contexto real. Isso pode fornecer informações valiosas sobre a aplicação prática do artefato.

Figura 1 – Quadro DSR



Fonte: adaptada de (PIMENTEL; FILIPPO; SANTORO, 2019)

## 1.4 Organização do Trabalho

Este trabalho está organizado da seguinte forma: No Capítulo 1, foi realizada uma introdução que contextualiza o trabalho e apresenta de forma sucinta os problemas, objetivos e metodologia adotada na pesquisa.

O Capítulo 2 dedica-se à fundamentação teórica essencial para a compreensão do estudo. Aborda tópicos como a acessibilidade *Web*, as diretrizes de acessibilidade do e-MAG, a ferramenta ASES para avaliação de acessibilidade e uma abordagem sobre *Web Crawlers*.

No Capítulo 3, são explorados trabalhos relacionados que oferecem um panorama das ferramentas utilizadas e os resultados alcançados em estudos similares.

No Capítulo 4, é apresentado o desenvolvimento do *Web Crawler* e do avaliador de acessibilidade. Detalha-se o funcionamento dessas duas partes centrais do trabalho, incluindo seus principais métodos.

O Capítulo 5 é descrito os testes realizados pelo *Web Crawler* e pelo avaliador de acessibilidade, abordando suas metodologias e resultados.

O Capítulo 6 traz as conclusões finais, destacando as contribuições proporcionadas pelo trabalho e delineando possíveis direções para futuras pesquisas na área.

## 2 Fundamentação Teórica

A fundamentação teórica deste trabalho está dividida em duas Seções. A primeira Seção aborda o conceito de acessibilidade *Web*, discutindo as recomendações de acessibilidade, incluindo o e-MAG, bem como a avaliação de acessibilidade, com destaque para a ferramenta ASES. A segunda Seção trata de conceitualizar o *Web Crawler*.

### 2.1 Acessibilidade *Web*

O princípio fundamental da acessibilidade *Web* é garantir que todos os serviços disponíveis *online* sejam acessíveis de forma universal. Isso implica que as páginas da *Web* não devem apresentar obstáculos que limitem ou dificultem o acesso ao conteúdo por parte dos usuários (WAI, 2005).

Além de ser benéfica para pessoas com deficiência, a acessibilidade *Web* também traz vantagens para idosos e indivíduos com limitações temporárias. Ademais, seguir boas práticas de acessibilidade *Web* melhora o acesso para usuários em diferentes situações, como ambientes com muita ou pouca iluminação, ambientes com ruído excessivo ou silenciosos, além de garantir compatibilidade com *hardwares* ou *softwares* antigos, entre outras circunstâncias (WAI, 2005).

Com o objetivo de promover a acessibilidade *Web*, o W3C, estabeleceu a *Web Accessibility Initiative* (WAI), uma iniciativa apoiada por diversos agentes envolvidos com a temática. Seu propósito é desenvolver padrões para os elementos essenciais, como navegadores, *players* de mídia, tecnologias assistivas e ferramentas de autoria, a fim de fornecer suporte e incentivar a acessibilidade *Web* (Web Accessibility Initiative (WAI), 2023).

As diretrizes e normas estabelecidas pela WAI são amplamente reconhecidas como padrões internacionais para assegurar a acessibilidade *Web*. Um exemplo paradigmático dessa abordagem é o (*Web Content Accessibility Guidelines* (WCAG)), a principal referência mundialmente reconhecida para acessibilidade *Web*. No entanto, é possível que cada país desenvolva suas próprias diretrizes de acessibilidade *Web* de acordo com suas necessidades específicas (WAI, 2005).

#### 2.1.1 Recomendações de acessibilidade *Web*

Com a crescente preocupação em garantir a acessibilidade digital para todos os cidadãos, alguns governos têm implementado políticas com o objetivo de assegurar que seus portais e serviços *online* sejam acessíveis a pessoas com deficiências. Como resultado, várias

diretrizes têm sido criadas para orientar a criação de páginas *Web* acessíveis (SLATIN; RUSH, 2003; BACH et al., 2009).

Nesse sentido o WCAG é um conjunto de recomendações desenvolvidas pelo W3C que visam tornar o conteúdo da *Web* mais acessível para pessoas com deficiência, incluindo visual, auditiva, física, de fala, intelectual, de linguagem, de aprendizagem e neurológica. O WCAG foi concebido para ser largamente aplicado às diferentes tecnologias atuais e futuras da *Web*. Sua diretrizes foram desenvolvida para serem testáveis com uma combinação de testes automáticos e avaliação humana (W3C, 2018).

Com a criação do WAI e o objetivo de viabilizar a implementação das Leis Federais nº 10.048 e nº 10.098, foi constituído um Comitê da Associação Brasileira de Normas Técnicas (ABNT) com a responsabilidade de comparar as normas de acessibilidade de diversos países e analisar as diretrizes propostas pelo W3C. Como resultado desses esforços, foi elaborado o e-MAG pelo Departamento de Governo Eletrônico, com o intuito de padronizar e facilitar o processo de tornar os *sites* acessíveis (SLATIN; RUSH, 2003; BACH et al., 2009).

O e-MAG é um importante compromisso do Governo Federal Brasileiro para promover a inclusão social por meio do acesso à informação digital. Sendo assim, o e-MAG atua como um guia para o desenvolvimento e adaptação de conteúdos digitais do governo federal, visando garantir que todos possam ter acesso às informações e serviços fornecidos pelo governo (Departamento de Governo Eletrônico, 2014).

As recomendações do e-MAG são desenvolvidas de forma padronizada, são fáceis de serem implementadas e coerentes com as necessidades específicas do Brasil, além de estarem em conformidade com os padrões internacionais. Atualmente, o e-MAG está na versão 3.1, que foi lançada em 2014. Essa nova versão contém 45 recomendações de acessibilidade, divididas em seis seções, como podemos observar na Figura 2: (1) Marcação, compreende 9 recomendações que oferecem diretrizes para a estruturação acessível do *site*; (2) comportamento, traz 7 recomendações que delineiam como uma página deve se comportar de tal maneira que o usuário tenha controle sobre os elementos do site, visando a acessibilidade para diferentes usuários; (3) conteúdo/informação, 12 recomendações que se relacionam com o conteúdo e a informação presente no *site* para que o usuário possa perceber as informações e compreendê-las; (4) apresentação/*design*, provê 4 recomendações que orientam aspectos de apresentação e *design* acessíveis também necessárias para compreensão do site.; (5) multimídia, incorpora 5 recomendações de vital importância para conteúdos de áudio, vídeo e animações.; (6) formulário, destaca 8 elementos cruciais para garantir a criação de formulários acessíveis (Departamento de Governo Eletrônico, 2014; SANTOS, 2023).

Figura 2 – Seções do e-MAG 3.1



Fonte: (SANTOS, 2023)

### 2.1.2 Avaliação de acessibilidade Web utilizando a ferramenta ASES

A avaliação da acessibilidade de um *site* pode ser realizada por meio de métodos como avaliação automatizada e manual. Durante a etapa de avaliação automatizada, são realizadas verificações utilizando ferramentas que analisam se o código *HyperText Markup Language* (HTML) estão em conformidade com recomendações de acessibilidade (CTA - Centro Tecnológico de Acessibilidade, 2018).

Nesse contexto, o ASES é uma ferramenta desenvolvida pelo governo brasileiro para avaliar páginas Web de acordo com o e-MAG. O ASES desempenha um papel crucial na avaliação da acessibilidade, produzindo um relatório que destaca os problemas identificados, conforme ilustrado na Figura 3. A porcentagem ASES, é uma métrica que indica o grau de conformidade da página podendo ser alto (acima de 95%), médio (de 85 a 94,99%), baixo (de 70 a 84,99%) e baixíssimo (abaixo de 70%). O Resumo de Acessibilidade por Seção eMAG apresenta a quantidade de erros e avisos detectados com base nas diretrizes do e-MAG. Os erros são elementos de acessibilidade detectáveis pela ferramenta e os avisos são possíveis problemas de acessibilidade que requerem uma avaliação humana (Portal do Software Público, 2017; CTA - Centro Tecnológico de Acessibilidade, 2018).

Figura 3 – Relatório de conformidade da página inicial da UFOP de acordo com o e-MAG

## Página Avaliada

Página: <http://ufop.br>

Título: Universidade Federal de Ouro Preto - UFOP

Tamanho: 86467 Bytes

Data/Hora: 04/07/2023 21:42:42

## Nota e Resumo da Avaliação de Acessibilidade



Legenda	
<span style="color: green;">■</span>	>= 95%
<span style="color: yellow;">■</span>	>= 85% < 95%
<span style="color: orange;">■</span>	>= 70% < 85%
<span style="color: red;">■</span>	< 70%

## Resumo de Acessibilidade por Seção eMAG

Seção	✖ Erro(s)	⚠ Aviso(s)
Marcação	23	126
Comportamento	1	6
Conteúdo/Informação	43	14
Apresentação / Design	1	0
Multimídia	0	0
Formulários	4	2
<b>Total</b>	<b>72</b>	<b>148</b>

Avaliação tem por base testes automáticos em código-fonte (X)HTML interpretados do [Modelo de Acessibilidade em Governo Eletrônico \(eMAG\)](#) ([link para novo site](#)).

A nota não contempla os itens classificados como avisos e aqueles que requerem avaliação humana. Para saber quais testes são contemplados pelo software, [favor verificar os critérios de sucesso trabalhados pelo ASESWEB](#).

Fonte: ([ASES, 2023](#))

Quando concluída a avaliação, é possível utilizar a funcionalidade de exportação disponível para gerar um documento em formato *Portable Document Format* (PDF). Esse documento conterá uma compilação organizada e formal de todos os resultados obtidos pelo avaliador apontando, além dos erros, os locais no código em. que eles ocorrem.. Através desta funcionalidade, é possível agregar de forma sistemática os dados e informações coletados durante todo o processo de avaliação. Ao exportar os resultados para um documento PDF, abre-se a possibilidade de compartilhar esses resultados com diversas partes interessadas.

Figura 4 – Resultado da avaliação em formato PDF

Avaliador e Simulador de Acessibilidade de Sítios

## ASES

GOVERNO FEDERAL

Relatório de Avaliação

**Página**

Página: http://ufop.br  
 Título: Universidade Federal de Ouro Preto - UFOP  
 Tamanho: 86467 Bytes  
 Data/ Hora: 04/07/2023 21:42:42

**Nota e Resumo da Avaliação de Acessibilidade**

Porcentagem	Seção	Erros	Avisos
ASES	Marcação	23	126
63.68%	Comportamento	1	6
	Conteúdo / Informação	43	14
	Apresentação / Design	1	0
	Multimídia	0	0
	Formulários	4	2
	<b>TOTAL</b>	<b>72</b>	<b>148</b>

**Detalhes da Avaliação**

**Marcação**

Erro	Recomendações	Quantidade	Linhas do Código Fonte
1.2 - Organizar o código HTML de forma lógica e semântica.		22	180, 181, 182, 213, 221, 222, 223, 381, 388, 395, 402, 409, 465, 484, 503, 522, 541, 560, 708, 718, 718, 775
1.3 - Utilizar corretamente os níveis de cabeçalho.		1	447

04/07/2023 Página 1 de 3

Fonte: (ASES, 2023)

Os avaliadores são úteis durante o processo de desenvolvimento de aplicações *Web* acessíveis, eles auxiliam na identificação de erros e avisos. É importante ressaltar o ASES testa um conjunto limitado de regras. Portanto, a ferramenta por si só não é capaz de detectar todos os problemas de acessibilidade de um *site*. Para uma avaliação completa, é necessário realizar análises manuais, que podem ser conduzidas por usuários reais, especialmente aqueles com deficiência, utilizando diferentes tecnologias, bem como por especialistas na área de acessibilidade *Web* (CTA - Centro Tecnológico de Acessibilidade, 2018).

## 2.2 Web Crawler

Um *Web Crawler* é um programa ou *script* que navega na *World Wide Web* (WWW) de forma automatizada e sistemática. Ele segue os *links* de uma página para outra, percorrendo um grafo direcionado que representa a estrutura da *Internet* como um navegador automatizado. Os *Web Crawlers* são também chamados de robôs, *spiders* ou *worms*. Eles são projetados para recuperar e armazenar páginas da *Web* em um repositório local. Essas páginas são posteriormente processadas por mecanismos de busca, que as

indexam para possibilitar a recuperação da informação de maneira rápida (KAUSAR; DHAKA; SINGH, 2013).

O funcionamento de um *Web Crawler* começa com um conjunto inicial de *Uniform Resource Locator* (URL) conhecidos como URL semente. Ele faz o *download* das páginas *Web* correspondentes às URL semente e extrai novos *links* presentes nessas páginas. As páginas *Web* recuperadas são armazenadas e indexadas em uma área de armazenamento para que possam ser posteriormente recuperadas quando necessário, utilizando esses índices. Esse processo é repetido até que não haja mais URL faltando para serem baixados (KAUSAR; DHAKA; SINGH, 2013).

Existem duas principais técnicas de rastreamento utilizadas por *Web Crawlers*:

- Rastreamento de propósito geral: Coleta o máximo de páginas possível de um conjunto de URL e seus *links*, abrangendo diferentes locais. Pode afetar a velocidade e a largura de banda da rede.
- Rastreamento focado: Coleta apenas documentos relacionados a um tópico específico, reduzindo o tráfego de rede e *downloads*. Rastreia seletivamente as regiões relevantes da *Web*, economizando recursos de *hardware* e rede.

Em resumo, um *Web Crawler* desempenha um papel fundamental na automação do processo *Web*, facilitando a recuperação eficiente de informações e dando suporte a várias aplicações na *Internet* (DHENAKARAN; SAMBANTHAN, 2011).



## 3 Trabalhos Relacionados

Este capítulo apresenta pesquisas que empregam os *Crawlers* para automatizar a avaliação de acessibilidade *Web* de um conjunto de páginas.

### 3.1 Observatório da Acessibilidade da *Web* Brasileira

([PARDINI et al., 2021](#)) abordam uma ferramenta composta por três módulos essenciais: o Coletor *Web*, que tem como propósito extrair todos os endereços de páginas de um *site* específico; o Avaliador Assíncrono, responsável pelo gerenciamento do processamento das páginas coletadas e pela utilização do *software* ASES para a avaliação da conformidade com o e-MAG; e o Portal, onde os resultados das avaliações são armazenados e apresentados.

O *software* desenvolvido coleta as páginas, priorizando uma abrangente cobertura e a otimização dos recursos computacionais. As páginas são submetidas a uma avaliação pelo ASES, fornecendo uma "nota de acessibilidade" e destacando os erros encontrados. Os resultados são então filtrados e armazenados para análises subsequentes no Portal.

A ferramenta adere a utilização exclusiva de conteúdo em HTML, não aceitando páginas que contenham Javascript. Atualmente, é utilizada a biblioteca **Scrapy** para realizar as coletas do *Crawler*, com planos futuros de migração para o mecanismo dinâmico DynWebStats e a incorporação de novos analisadores, a exemplo do WCAG.

Um estudo avaliou de aproximadamente 8 mil *sites*. O Portal oferece uma visão panorâmica dos resultados, incluindo faixas de conformidade e listagens de *sites* por Unidades Federativas. Destaca-se a seção "Erros Mais Comuns", que sumariza informações sobre as recomendações do e-MAG, suas descrições e os totais de erros encontrados, tanto por *sites* quanto por páginas como podemos ver na Figura 5.

Figura 5 – Listagem dos Erros Mais Comuns, com destaque para a seção de erros do tipo Marcação

Número	Descrição	Total de Erros	Total de Sites	Total de Páginas
<b>1.1</b>	<b>Respeitar os Padrões Web</b>	7.241.217	415	227.140
<b>1.2</b>	<b>Organizar o código HTML de forma lógica e semântica</b>	1.381.146	434	228.903
<b>1.3</b>	<b>Utilizar corretamente os níveis de cabeçalho</b>	430.272	450	229.368
<b>1.5</b>	<b>Fornecer âncoras para ir direto a um bloco de conteúdo</b>	440.203	437	225.795
<b>1.6</b>	<b>Não utilizar tabelas para diagramação</b>	6.721	76	37.756
<b>1.7</b>	<b>Separar links adjacentes</b>	325.269	195	164.160

Fonte: (PARDINI et al., 2021)

### 3.2 *Towards dDistribution of Web Sites in a Crawler Used for Large-Scale Web Accessibility Assessment*

Neste estudo feito por CHEN, WANG e OLSEN, explora-se a criação de um sistema para avaliar extensivamente o conteúdo *online*, como uma ferramenta para medir a acessibilidade de páginas *Web* ou como um motor de busca aprimorado. O cerne desse sistema é um *Web Crawler*, um agente digital, que pode ser operado de forma distribuída. O objetivo é ampliar a capacidade do *Crawler* para baixar versões atualizadas de *WebSites*, quando comparadas às cópias armazenadas localmente, enquanto também se minimiza o tempo necessário para analisar toda a gama de *sites*. Esse desafio é tratado como um dilema de distribuição de recursos, pois o *Crawler* precisa eficientemente balancear o tempo dedicado a baixar e avaliar *sites* com os recursos limitados disponíveis, que variam de um ponto de acesso para outro.

Esse estudo propõe uma nova abordagem para esse enigma de alocação de recursos, preenchendo uma lacuna na pesquisa existente, que muitas vezes se concentra em soluções não-distribuídas ou estratégias de amostragem. Os pesquisadores estendem o algoritmo OMA e introduzem uma versão ponderada pelo tempo do algoritmo, conhecida como TWOMA. Resultados experimentais ilustram que o TWOMA é eficaz em substancialmente reduzir o tempo necessário para coletar e avaliar até 10.000 *sites Web*, quando empregado em conjunto com um *Web Crawler* distribuído. Essa abordagem supera os métodos tradicionais de distribuição uniforme de *sites*. Notavelmente, o TWOMA é capaz de reduzir

pela metade o tempo de cada coleta de dados, conforme demonstrado nos ambientes experimentais.

### 3.3 *Monitoring Accessibility of Governmental Websites in Europe*

Este projeto de (BÜHLER et al., 2008) aborda a acessibilidade *Web* como um objetivo importante da estratégia europeia i2010. O estudo apresenta uma abordagem para complementar os resultados de pesquisas pontuais de acessibilidade *Web* com avaliações automatizadas, que podem ser repetidas regularmente. O projeto é baseado na *European Internet Accessibility Observatory* (EIAO), que oferece uma base de *software* para a análise. O objetivo é comparar os dados coletados pela EIAO com outros estudos, visando fornecer uma visão abrangente da acessibilidade *Web* na Europa.

A Comissão Europeia reconheceu a importância da inclusão digital e adotou estratégias como eEurope e 2010, que ressaltam a acessibilidade na aquisição de tecnologias da informação, certificação de acessibilidade e uso da legislação. A EIAO, por meio de seu observatório de acessibilidade na *internet*, utiliza um rastreador *Web* para identificar recursos disponíveis, um repositório de URL, métricas de acessibilidade e um armazenamento de dados. A análise comparativa com outros estudos demonstrou que a EIAO oferece informações complementares confiáveis, sendo que as avaliações manuais e automatizadas estão alinhadas, permitindo monitoramento contínuo e análise evolutiva da acessibilidade *Web* europeia.

## 4 Desenvolvimento

Este capítulo apresenta a criação do artefato AccessCrawler um programa desenvolvido em *Python* que se destina à avaliação dos *links* pertencentes ao domínio da UFOP por meio da ferramenta ASES. O programa consiste em dois componentes essenciais, um *Web Crawler* e um Avaliador. O projeto tem a capacidade de identificar todos os *links* disponíveis e avaliá-los utilizando a ferramenta do ASES. Ele realiza a verificação para determinar se um *link* é válido para a avaliação ou se contém erros, armazenando as informações resultantes em conjuntos distintos, com a finalidade de análise subsequente.

O *Web Crawler* assume a responsabilidade inicial de varrer o domínio da UFOP em busca de *links*. Essa tarefa, apesar de parecer simples, é fundamental para a compreensão das páginas e recursos associados a esse domínio. Com essa base sólida de informações, o próximo passo se concentra em avaliar a acessibilidade desses *links* por meio do Avaliador.

Para otimizar esse processo, os conjuntos são uma estrutura escolhida para armazenar os *links* coletados. Eles oferecem a capacidade de representar coleções não ordenadas e a garantia de elementos únicos e imutáveis. Essa abordagem não só melhora a manipulação dos dados, mas também otimiza a verificação de pertinência dos elementos, fundamental para a validação dos *links* encontrados.

No cerne da avaliação dos *links* a implementação de expressões regulares, permite a identificação de extensões indesejadas e palavras-chave associadas à autenticação ou redes sociais. Dessa maneira, os *links* que não atendem a esses critérios são automaticamente descartados, direcionando a atenção apenas para aqueles de interesse real.

Além disso, os *links* válidos também são submetidos a critérios específicos: eles devem conter o domínio da UFOP e iniciar com "http://" ou "https://". Essa estratégia metódica permite que o programa se concentre exclusivamente em *links* pertinentes e relevantes, evitando a alocação desnecessária de recursos em *links* que não atendem aos critérios estabelecidos. A seção de filtros, apresentará informações detalhadas sobre esses critérios adotados para determinar os *links* válidos durante o processo de verificação. Serão exploradas as expressões regulares utilizadas para identificar e filtrar *links* indesejados, abordando aspectos como extensões de arquivos específicas, palavras-chave, autenticação e *links* de redes sociais, entre outros.

Além disso, para aumentar a eficiência do processo de verificação, a utilização de *threads* é adotada. Essa abordagem possibilita a análise simultânea de várias páginas, resultando em um uso mais eficaz dos recursos e otimização do tempo de processamento.

A estrutura do projeto que pode ser visualizada no Código 4.1 compreende os ar-

quivos `main.py`, `arquivos.py`, `processar_links.py`, `encontrar_links.py` e `filtro.py`, que contêm as funções e módulos essenciais para a execução do programa. O diretório `arquivosTXT` armazena os arquivos de texto que contêm os conjuntos de *links* encontrados, processados e com erro, além do arquivo `html_link_verificado.txt`, que guarda o HTML da página onde um *link* específico foi encontrado e o `avaliacao.txt` que apresenta o resultado da avaliação de cada *link*. Segue abaixo a estrutura:

Código 4.1 – Estrutura de diretórios e arquivos utilizada no projeto.

```
1 - main.py
2 - arquivos.py
3 - processar_links.py
4 - encontrar_links.py
5 - filtro.py
6 - avaliador.py
7 - arquivosTXT/
8   - links_encontrados.txt
9   - links_processados.txt
10  - links_com_erro.txt
11  - html_link_verificado.txt
12  - avaliacao.txt
```

Fonte: Elaborado pelo autor, 2023

Antes de executar o programa, é necessário realizar algumas configurações iniciais no arquivo `main.py`. Ele contém as variáveis `link_inicial` e `dominio`, que definem o *link* a partir do qual o processamento será iniciado e o domínio a ser analisado, respectivamente.

Após executar o *Crawler* e obter todos os *links* que serão avaliados, é o momento de iniciar a avaliação de acessibilidade dos *links*. Para essa tarefa, utilizaremos o arquivo `avaliador.py`, que também utiliza uma abordagem de processamento paralelo com *threads* para avaliar simultaneamente múltiplos *links*, otimizando o tempo de execução.

O programa extrai informações e registra esses dados no arquivo `avaliacao.txt`. Após cada avaliação, os resultados são registrados nesse arquivo em linhas, sendo cada uma referente a um *link* avaliado como podemos ver na Figura 6. Os valores separados por ponto e vírgula são os valores da avaliação. Cada campo representa informações específicas sobre o *link* analisado, tais como o próprio *link*, a porcentagem de acessibilidade obtida na avaliação, o total de erros e o total de avisos encontrados. Além disso, os campos subsequentes referem-se aos erros e avisos específicos das seções do e-MAG: Marcação, Comportamento, Conteúdo e Informação, Apresentação e *Design*, Multimídia e Formulários.

Figura 6 – Avaliação dos *links* no arquivo *avaliacao.txt*

```

3 2023-08-11 18:28:26.449521;https://proplad.ufop.br/%3Cnolink%3E/coordenadorias-planejamento/coordenadoria-de-avaliacao-permanente;78,77,9;50;4;41;1;3;0;4;0;0;0;4;2;
4 2023-08-11 18:28:26.446524;http://nel.ufop.br/index.php?option=com_content&view=category&id=59&Itemid=41&limitstart=20;91,05;11;102;6;92;0;4;5;5;0;0;0;0;1;
5 2023-08-11 18:28:26.450543;https://www.em.ufop.br/index.php/resolucoes/2-uncategorised/318-resolucoes-cdm-sobre-cepe-7000-2018;85,52;8;63;5;58;0;1;2;4;1;0;0;0;0;0;
6 2023-08-11 18:29:00.736603;https://vestibular.ufop.br/index.php?option=com_content&view=category&id=52&Itemid=149;91,01;6;23;4;18;1;1;1;3;0;0;0;0;1;
7 2023-08-11 18:28:26.450543;https://propp.ufop.br/en/node/690;66,02;46;94;9;72;1;2;31;18;1;0;0;0;4;2;
8 2023-08-11 18:28:26.447525;https://propp.ufop.br/en/situacao/encerrado;69,68;16;85;5;75;1;2;6;6;0;0;0;0;4;2;
9 2023-08-11 18:29:05.265164;http://www3.decom.ufop.br/toffolo/pt-br/admin/?next=;87,93;5;10;4;9;0;0;0;0;1;0;0;0;0;1;
10 2023-08-11 18:28:26.451523;https://www.caint.ufop.br/oportunidades/ed-encerrados;79,33;24;96;7;84;0;4;15;3;1;0;0;0;1;5;
11 2023-08-11 18:28:26.452521;https://www.dri.ufop.br/see/1118-ets-possibilita-a-realizacao-do-toefl-ibt-e-gre-general-test-em-casa;77,35;24;111;7;100;0;4;16;4;1;0;0;0;0;3;
12 2023-08-11 18:29:06.172759;https://propp.ufop.br/en/node/370;63,58;18;61;6;50;1;2;6;7;1;0;0;0;4;2;
13 2023-08-11 18:29:00.698166;https://decgp.medicina.ufop.br/notK3%ADcias-1;82,93;10;46;6;34;1;3;2;8;1;0;0;0;0;1;
14 2023-08-11 18:29:07.674226;https://www.cppo.ufop.br/gestao-da-integridade;78,75;10;79;4;59;1;3;1;15;0;0;0;0;4;2;
15 2023-08-11 18:29:10.776842;http://www.aceessoainformacao.ufop.br/index.php/2012-04-10-18-20-19/quem-e-quem/18-nucleo-de-tecnologia-da-informacao;75,9;18;50;14;44;1;6;3;0;0;0;0;0;0;
16 2023-08-11 18:29:03.715903;http://dri.ufop.br/see/1515-universidad-del-rosario-colombia-convida-para-palestra-get-to-know-universidad-del-rosario;77,36;23;108;7;98;0;4;15;3;1;0;0;0;0;3;
17 2023-08-11 18:28:26.448547;http://xr4goodlab.decom.ufop.br/?m=201907;74,12;54;161;15;125;1;8;36;26;0;0;0;0;2;2;
18 2023-08-11 18:29:10.057907;http://prace.ufop.br/restaurante-universitario/ru-para-visitantes;69,65;19;119;7;106;1;4;7;7;0;0;0;0;4;2;
19 2023-08-11 18:28:26.451523;https://www.decom.ufop.br/workshop/2013/.../cacic;80,96;54;123;33;95;1;7;20;16;0;5;0;0;0;0;
20 2023-08-11 18:29:13.471495;https://fimat.ufop.br/file/65549;73,1;12;34;4;26;1;4;6;4;1;0;0;0;0;0;
21 2023-08-11 18:29:21.997110;http://www.processoseletivo.ufop.br/index.php?option=com_content&view=category&id=35&Itemid=162&limitstart=15;91,01;6;23;4;18;1;1;1;3;0;0;0;0;1;
22 2023-08-11 18:29:22.906771;https://www.propp.ufop.br/en/node/693;66;47;87;9;70;1;2;32;13;1;0;0;0;4;2;
23 2023-08-11 18:29:18.357607;http://xr4goodlab.decom.ufop.br/?page_id=6296;74,95;20;195;11;177;1;8;6;8;0;0;0;0;2;2;
24 2023-08-11 18:29:25.621049;https://prox.ufop.br/documentos/formularios-modelos-e-guias;65,35;20;91;7;80;1;5;8;4;0;0;0;0;4;2;
25 2023-08-11 18:28:26.453524;http://www.caint.ufop.br/read/1751-gcub-oferece-desconto-na-realizacao-do-exame-toefl-ibt-12;77,24;28;131;0;119;0;5;18;4;1;0;0;0;0;3;
26 2023-08-11 18:29:05.748029;https://www.dri.ufop.br/en/postgraduate-programs;77,98;14;141;5;130;0;4;8;4;1;0;0;0;0;3;
27 2023-08-11 18:29:36.732574;https://vestibular.ufop.br/index.php?option=com_content&view=category&id=42&Itemid=139;91,01;6;23;4;18;1;1;1;3;0;0;0;0;1;
28 2023-08-11 18:29:38.117385;https://sisbin.ufop.br/bibliotecas/biblioteca-icsa;67,6;18;43;4;33;1;4;8;4;1;0;0;0;4;2;
29 2023-08-11 18:29:35.064747;https://residencia.medicina.ufop.br/c1K3%ADnica-mNC3%ADnica;78,75;9;45;6;33;1;3;1;9;1;0;0;0;0;0;
30 2023-08-11 18:29:27.852648;https://www.prace.ufop.br/assistencia-estudantil/orientacao-estudantil/programa-de-incentivo-diversidade-e-convivenciapidic;65,56;23;124;6;107;1;4;12;11;0;0;0;0;4;2;

```

Fonte: Elaborado pelo autor, 2023

Ao reunir e consolidar todas as avaliações em um único arquivo, é possível obter uma visão geral das métricas de acessibilidade do *site* da UFOP, permitindo à equipe responsável tomar decisões informadas e implementar melhorias de forma estratégica. O arquivo *avaliacao.txt* torna-se uma referência essencial para monitorar a evolução da acessibilidade ao longo do tempo, servindo como base para ações contínuas de aprimoramento e garantindo a conformidade com as diretrizes de acessibilidade estabelecidas.

## 4.1 Estrutura do Código

A seguir, os códigos acompanhados de suas respectivas funcionalidades de forma detalhada.

### 4.1.1 Main.py

O Código 4.2 coordena o fluxo de execução, sendo responsável pela leitura dos arquivos de texto e pela atribuição dos valores aos conjuntos `links_encontrados`, `links_processados` e `links_com_erro`. Caso a variável `links_encontrados` esteja vazia, o conjunto é inicializado com o *link* inicial fornecido, esse mecanismo foi implementado para o caso de o programa parar a execução, o progresso não seja perdido. Em seguida, a função `inicia_threads()`, presente no módulo `processar_links` na linha 21, é invocada, passando os conjuntos de *links* como argumentos. Esta função é responsável por iniciar o processamento dos *links* em *threads* separadas, o que permite uma execução mais eficiente e paralela. Após o processamento realizado pelas *threads*, os arquivos de texto são atualizados com os novos *links* encontrados.

As variáveis globais `links_encontrados`, `links_processados`, `links_com_erro`, `link_inicial` e `link_a_verificar` são utilizadas para manter o controle e o estado

dos *links* durante a execução do código. Elas são atualizadas conforme novos *links* são encontrados, processados ou apresentam erros.

Código 4.2 – Método principal do programa.

```
1 links_encontrados = set()
2 links_processados = set()
3 links_com_erro = set()
4
5 link_inicial = 'https://www.ufop.br/'
6 dominio = 'ufop'
7 link_a_verificar = ''
8
9 def main():
10     while(True):
11         links_encontrados = arquivos.ler_linhas_arquivo(links_encontrados, "arquivosTXT/links_encontrados.txt")
12         links_processados = arquivos.ler_linhas_arquivo(links_processados, "arquivosTXT/links_processados.txt")
13         links_com_erro = arquivos.ler_linhas_arquivo(links_com_erro, "arquivosTXT/links_com_erro.txt")
14
15         if len(links_encontrados) == 0 and len(links_processados) == 0:
16             links_encontrados.add(link_inicial)
17         elif len(links_encontrados) == 0:
18             print("Chegamos ao fim!!")
19             break
20
21         links_encontrados, links_processados, links_com_erro = processar_links.inicia_threads(links_encontrados, links_processados, links_com_erro, link_a_verificar)
22
23         arquivos.escrever_linhas_arquivo(links_encontrados, "arquivosTXT/links_encontrados.txt")
24         arquivos.escrever_linhas_arquivo(links_processados, "arquivosTXT/links_processados.txt")
25         arquivos.escrever_linhas_arquivo(links_com_erro, "arquivosTXT/links_com_erro.txt")
26     )
```

Fonte: Elaborado pelo autor, 2023

### 4.1.2 Arquivos.py

No trecho de código do `arquivos.py`, as funções `ler_linhas_arquivo()`, `escrever_linhas_arquivo()` e `remover_linhas_arquivo()` têm como propósito simplificar a leitura, escrita e remoção de *links* de arquivos, facilitando o processamento e a atualização das informações durante a execução do *script* principal.

A função `ler_linhas_arquivo()` é responsável por ler as linhas de um arquivo especificado, realizando uma operação de limpeza para remover espaços em branco e quebras de linha. Os *links* encontrados nas linhas são então atualizados em um conjunto e retornados como resultado da função.

A função `escrever_linhas_arquivo()` escreve os *links* de um conjunto em um

arquivo específico, substituindo os *links* anteriores presentes no arquivo. Cada *link* é escrito seguido por uma quebra de linha para organização.

A função `remover_linhas_arquivo()` remove *links* específicos de um arquivo. Ela lê as linhas do arquivo, calcula a diferença entre os *links* lidos e os *links* a serem removidos e, em seguida, atualiza o arquivo sem os *links* removidos.

Essas funções desempenham um papel crucial no programa, simplificando a manipulação e atualização de *links* armazenados em arquivos durante a execução do código.

### 4.1.3 Processar\_links.py

Neste trecho de código, é realizada a coordenação do processamento dos *links* encontrados, incluindo a iniciação das *threads* de processamento como podemos ver no Código 4.3 e a atualização dos conjuntos `links_processados` e `links_com_erro` com base nos resultados visualizado no Código 4.4.

A função `inicia_threads` desempenha um papel crucial na orquestração do processamento dos *links* durante a execução do programa. O procedimento ocorre por meio de uma iteração sobre os *links* encontrados, onde uma *thread* é criada para processar cada um deles e é adicionada à lista `threads`. Após todas as *threads* serem iniciadas, a função aguarda a conclusão da execução de cada uma, garantindo o processamento completo dos *links*. Ao finalizar o processamento, a função atualiza o conjunto `links_encontrados` com os novos *links* encontrados durante o rastreamento. Como resultado, a função retorna uma tupla contendo os conjuntos `links_encontrados`, `links_processados` e `links_com_erro` atualizados.

Código 4.3 – Inicialização das *threads* e retorno dos *links* encontrados.

```
1  for link in links_encontrados:
2      thread = Thread(target=processar_link, args=(novos_links_encontrados, link,
3          link_a_verificar))
4      thread.start()
5      threads.append(thread)
6
7  # Espera tudo terminar
8  for thread in threads:
9      thread.join()
10
11  for link in novos_links_encontrados:
12      if link not in links_processados and link not in links_com_erro:
13          links_encontrados.add(link)
14
15  return links_encontrados, links_processados, links_com_erro
```

Fonte: Elaborado pelo autor, 2023

Por sua vez, a função `processar_link` desempenha um papel específico no processamento individual de cada *link* encontrado. Seus parâmetros incluem uma lista denominada



`results`, que será usada para armazenar os novos *links* encontrados durante o processamento e o *link* a ser processado.

O procedimento começa com verificando se o *link* ainda não foi processado (*link* não está presente em `links_processados`) e também se não consta na lista de *links* com erro (*link* não está em `links_com_erro`). Se o *link* ainda não foi processado, a função invoca `encontrar_links.encontrar_links()` do módulo `encontrar_links` na linha 3, fornecendo os conjuntos `links_processados`, `links_com_erro` e o *link* a ser processado.

Se `encontrar_links()` retornar `True` (indicando que o *link* foi processado com sucesso e novos *links* foram encontrados), o *link* é adicionado a `links_processados`, e os novos *links* encontrados são adicionados à lista `results`. Se `encontrar_links()` retornar `False` (indicando que o *link* não foi encontrado), o *link* é adicionado a `links_com_erro`.

Código 4.4 – Processamento dos *links*.

```
1  if link not in links_processados and link not in links_com_erro:
2
3      novos_links_encontrados, encontrou = encontrar_links.encontrar_links(
4          links_processados, links_com_erro, link)
5
6      if(encontrou):
7          links_processados.add(link)
8          results.extend(novos_links_encontrados)
9
10     else:
11         links_com_erro.add(link)
12
13 else:
14     print("LINK JA PROCESSADO!", link)
```

Fonte: Elaborado pelo autor, 2023

#### 4.1.4 Encontrar\_links.py

A função `encontrar_links` desempenha um papel essencial no contexto do programa, sendo responsável por realizar a busca de *links* em uma página da *Web*.

No cerne deste processo, a Seção do Código 4.5 assume a tarefa de localizar os *links* na página, a partir da URL fornecida. A implementação utiliza a biblioteca `requests` para realizar uma solicitação *HyperText Transfer Protocol* (HTTP) GET à página especificada utilizando o método `requests.get()`. Caso a solicitação seja bem-sucedida, com um código de *status* 200, o conteúdo HTML da página é então submetido à análise com a biblioteca `BeautifulSoup`, onde é empregada para analisar o conteúdo HTML e extrair os *links*. Adicionalmente, a função `filtro.valida_link()` é aplicada para validar cada *link* encontrado, antes de adicioná-lo ao conjunto `links_encontrados`.

Código 4.5 – Método encontrar *links*.

```
1  try:
2      response = requests.get(url, verify=False)
```

```

3     except Exception as err:
4         print("Erro no Link: " + url)
5         return links_encontrados, False
6
7     if response.status_code == 200:
8         conteudo_html = response.text
9         soup = BeautifulSoup(conteudo_html, 'html.parser')
10        links = soup.find_all('a')
11        for a in links:
12            link = a.get('href')
13            link = verificar_link_local(url, link)
14
15            if filtro.valida_link(links_processados, links_com_erro, link):
16                links_encontrados.add(link)
17
18        return links_encontrados, True
19    else:
20        print('Erro ao acessar a página:', response.status_code, ' | ', url)
21        return links_encontrados, False

```

Fonte: Elaborado pelo autor, 2023

Posteriormente, os elementos de âncora presentes na página são identificados e os URLs contidos neles são extraídos. Para cada URL extraído, invoca-se a função `verificar_link_local(url, link)` a fim de verificar se é um *link* local ou relativo, convertendo-o para um formato absoluto, se necessário, utilizando o método do Código 4.6. Nessa função, verifica-se primeiramente se o *link* inicia com 'http'. Caso o *link* se inicie com 'http', é considerado um URL absoluto, ou seja, um *link* que aponta diretamente para um recurso em um domínio externo. Por outro lado, se o *link* iniciar com '/', trata-se de um *link* relativo à raiz do domínio, o qual é convertido para um URL absoluto verificado. Dessa forma, os *links* relativos são transformados em *links* absolutos, garantindo que os navegadores possam acessá-los corretamente.

Código 4.6 – Gerar *links* absoluto.

```

1     if link.startswith('http'):
2         return link
3
4     if link.startswith('/'):
5
6         dominio_url = url
7         posicao_barra = url.find("/", 8)
8         if posicao_barra != -1:
9             dominio_url = url[:posicao_barra]
10
11        return dominio_url + link

```

Fonte: Elaborado pelo autor, 2023

Antes de adicionar cada *link* ao conjunto `links_encontrados`, sua validade é verificada por meio da função `filtro.valida_link()` do módulo `filtro` na linha 15 do Código 4.5. Somente *links* válidos são incorporados ao conjunto. Ao concluir a validação

de todos os *links*, a função retorna uma tupla contendo o conjunto de *links* encontrados e um valor booleano que indica se o *link* foi processado ou não.

As funções `encontrar_links` e `verificar_link_local` trabalham em conjunto para efetuar a verificação e o processamento adequado dos *links* em uma página da *Web*, desempenhando um papel fundamental no fluxo de execução do programa.

#### 4.1.5 Filtro.py

Esta seção do Código 4.7 é responsável por validar os *links* encontrados antes de adicioná-los aos conjuntos `links_processados` ou `links_com_erro`. A função utiliza expressões regulares para verificar se o *link* possui características válidas ou indesejadas, garantindo que apenas os *links* adequados sejam processados.

Código 4.7 – Filtros de validação de *links*.

```
1  if regex_ufop.search(link) and regex_http_https.search(link):
2      if not (regex_extensoes.search(link)
3          or regex_palavras.search(link)
4          or regex_barra_dupla.search(link)
5          or regex_autenticacao.search(link)
6          or regex_absurdos.search(link)
7          or regex_openScholar.search(link)
8          or regex_rede_social.search(link)
9          or regex_downloads.search(link)):
10
11         if link not in links_processados and link not in links_com_erro:
12             return True
13
14  return False
```

Fonte: Elaborado pelo autor, 2023

A função `valida_link` é responsável por validar os *link*. Inicialmente, a função determina várias expressões regulares, definidas para realizar a verificação do *link*. Essas expressões têm o propósito de identificar partes indesejadas ou características específicas que indicariam que o *link* não é válido como por exemplo `.css`, `.gif`, `.js`, `.png`. Além disso, outras expressões regulares são utilizadas para verificar se o *link* contém partes válidas, como ter o trecho `.ufop` no *links*, *links* HTTP ou *HyperText Transfer Protocol Secure* ([HTTPS](#)).

Em seguida, algumas regras que validam *links* válidos: (1) `regex_ufop`, verifica se o *link* contém o domínio da UFOP, definido pela variável `main.dominio`. Isso garante que os *links* sejam específicos para a UFOP; (2) `regex_http_https`, busca por *links* que comecem com `http://` ou `https://`, indicando que são *links* HTTP ou HTTPS, respectivamente.

Se o *link* atender a todas as regras de *links* válidos, ele é adicionado ao conjunto `links_processados` e a função retorna `True`. Caso contrário, o *link* é adicionado ao conjunto `links_com_erro` e a função retorna `False`. Dessa forma, apenas os *links* válidos

que ainda não foram processados ou que não estão na lista de erros serão adicionados ao conjunto `links_encontrados`, garantindo uma coleção de *links* válidos e únicos para posterior análise. Essas regras e validações são essenciais para assegurar a qualidade e precisão dos resultados obtidos durante o processamento dos *links* no contexto do programa.

A criação de uma expressão regular chamado (`regex_absurdos`) é opcional, foi implementado visando otimizar o desempenho e reduzir a coleta de *links* que serão encontrados durante a execução do programa. Essa abordagem permite uma coleta mais focada e direcionada, excluindo *links* que provavelmente não são relevantes para a análise de acessibilidade como por exemplo *links* de calendários que para cada dia do ano cria um *link* novo. No entanto, é importante ressaltar que, caso a intenção seja realizar uma coleta mais abrangente e incluir todos os *links*, é essencial remover o uso da expressão regular de palavras "absurdas" para garantir que nenhum *link* seja inadvertidamente excluído. A escolha de usar ou não essa expressão regular dependerá dos objetivos específicos da análise e das necessidades do projeto.

#### 4.1.6 Avaliador.py

A função principal do Código 4.8 coordena o fluxo de execução. Para isso, ele lê os *links* processados de um arquivo de texto utilizando a função `arquivos.ler_linhas_arquivo()`, e em seguida, cria um conjunto de *threads* usando `ThreadPoolExecutor`. Cada *thread* é atribuída à função de avaliação `avaliarLink(link)`, com um *link* específico como argumento. O número máximo de *threads* simultâneas é definido como 10 por conta dos recursos disponíveis na máquina de teste. Ao final da avaliação, uma mensagem de conclusão é exibida no *console*.

Código 4.8 – Processo de avaliação.

```
1 def avaliarLink(link):
2     avaliacao = Avaliacao(link)
3     navegador = getNavegador()
4     getAvaliacao(navegador, link)
5     getResposta(navegador, avaliacao)
6     liberarNavegador(navegador)
7     pass
```

Fonte: Elaborado pelo autor, 2023

A função `avaliarLink()` é responsável por avaliar um *link* específico passado como parâmetro. Dentro dessa função, uma instância da classe `Avaliacao` que pode ser visualizada no Código 4.9 é criada para representar os resultados da avaliação de um *link* específico. Essa classe possui um construtor `init` que recebe o parâmetro `link`, representando o *link* da página avaliada.

No construtor, são inicializadas diversas variáveis de instância, cada uma representando uma característica específica da avaliação. Estas variáveis incluem `titulo`,

porcentagem, data, tamanho, tempoAvaliacao, secoes, erros e avisos. A maioria delas é inicializada com valores vazios ou listas vazias, exceto `link`, que recebe o valor passado como argumento.

A classe também possui um método chamado `to_string()`, que retorna uma representação em formato de *string* dos resultados da avaliação. Este método é utilizado para criar uma linha de texto que será escrita em um arquivo contendo as informações de avaliação.

O método `to_string()` percorre as listas `erros` e `avisos`, concatenando seus elementos separados por ponto e vírgula. Isso garante que todas as informações de `erros` e `avisos` sejam apresentadas em uma única linha, facilitando a leitura e a gravação em um arquivo. O método retorna uma *string* no formato "link;porcentagem;erros;avisos", em que `erros` e `avisos` são as listas concatenadas.

Código 4.9 – Classe Avaliação.

```
1 class Avaliacao:
2
3     def __init__(self, link):
4
5         self.link = link
6         self.titulo = ""
7         self.porcentagem = ""
8         self.data = ""
9         self.tamanho = ""
10        self.tempoAvaliacao = ""
11        self.secoes = [""]
12        self.erros = []
13        self.avisos = []
14
15    def to_string(self):
16        erros_e_avisos = ""
17        max_length = max(len(self.erros), len(self.avisos))
18
19        for i in range(max_length):
20            if i < len(self.erros):
21                erros_e_avisos += self.erros[i] + ";"
22            if i < len(self.avisos):
23                erros_e_avisos += self.avisos[i] + ";"
24
25        return self.link + ";" + self.porcentagem + ";" + erros_e_avisos.strip()
```

Fonte: Elaborado pelo autor, 2023

Para realizar a avaliação de um *link*, a função obtém um navegador disponível usando `getNavegador()` disponível no Código 4.10. Se não houver navegadores disponíveis na lista `navegadores_disponiveis`, uma nova instância do navegador Firefox é criada e configurada para funcionar em modo *headless* (sem abrir visualmente o navegador). Caso contrário, um navegador disponível é retirado da lista.

Após obter o navegador, a função `getAvaliacao()` é chamada para abrir a página

"https://asesweb.governoeletronico.gov.br/" no navegador e inserir o *link* a ser avaliado no campo de entrada de URL da página. Em seguida, é executada a avaliação do *link* clicando no botão "Executar".

Código 4.10 – Método de inicialização dos navegadores.

```
1 def getNavegador():
2     if not navegadores_disponiveis:
3         service = Service(GeckoDriverManager().install())
4         firefox_options = Options()
5         firefox_options.add_argument('-headless')
6         navegador = webdriver.Firefox(service=service, options=firefox_options)
7     else:
8         navegador = navegadores_disponiveis.pop()
9     return navegador
10
11 def liberarNavegador(navegador):
12     navegadores_disponiveis.append(navegador)
```

Fonte: Elaborado pelo autor, 2023

Após a avaliação, a função `getResposta()` disponível no Código 4.11 é chamada para obter os resultados da avaliação verificando se o elemento da porcentagem da avaliação está presente na página usando a função `awaitElemento(navegador)`. Essa foi a forma identificada para garantir que a avaliação já foi finalizada, e os dados já estão presentes na tela. Essa função utiliza a classe `WebDriverWait` do Selenium para esperar por um elemento específico na página do navegador dentro de um tempo limite de 70 segundos. Se o elemento for encontrado dentro desse tempo, as funções `getPorcentagem()` e `getTabela()` são chamadas para obter os resultados da avaliação e preencher a instância de `Avaliacao`, quando finalizadas, o método `setResposta()` é chamado para salvar os resultados.

Código 4.11 – Processo de avaliação.

```
1 def getResposta(navegador, avaliacao: Avaliacao):
2
3     if awaitElemento(navegador):
4         getPorcentagem(navegador, avaliacao)
5         getTabela(navegador, avaliacao)
6         setResposta(avaliacao)
7         return True
8
9     return False
```

Fonte: Elaborado pelo autor, 2023

O método `getPorcentagem()` é responsável por obter a porcentagem de acessibilidade do *link* avaliado na página de avaliação visto na Figura 7. Já o método `getTabela()` é responsável por obter a tabela de erros e avisos associada ao *link* avaliado na página de avaliação visto na Figura 8.

Figura 7 – Porcentagem de avaliação



Fonte: (ASES, 2023)

Figura 8 – Tabela de Avisos e Erros

#### Resumo de Acessibilidade por Seção eMAG

Seção	✖ Erro(s)	⚠ Aviso(s)
Marcação	24	128
Comportamento	1	6
Conteúdo/Informação	50	16
Apresentação / Design	1	0
Multimídia	0	0
Formulários	4	2
<b>Total</b>	<b>80</b>	<b>152</b>

Fonte: (ASES, 2023)

Os dados da tabela obtidos pelos métodos `getTabela()` estão divididos em duas seções: erros e avisos, referentes a diferentes categorias de avaliação de acessibilidade. Cada categoria representa aspectos específicos da avaliação de acessibilidade do *link*, sendo elas: (1) Erros de Marcação; (2) Avisos de Marcação; (3) Erros de Comportamento; (4) Avisos de Comportamento; (5) Erros de Conteúdo e Informação; (6) Avisos de Conteúdo e Informação; (7) Erros de Apresentação e *Design*; (8) Avisos de Apresentação e *Design*; (9) Erros de Multimídia; (10) Avisos de Multimídia; (11) Erros de Formulários; (12) Avisos de Formulários.

O método `setResposta(avaliacao)` é responsável por escrever os resultados da avaliação em um arquivo de texto. Ele recebe como parâmetro uma instância da classe `Avaliacao`, que contém todas as informações sobre a avaliação de um *link* específico.

Ao chamar esse método, ele entra em uma seção crítica protegida por um `lock` para garantir que apenas uma *thread* de cada vez possa escrever no arquivo compartilhado. Isso evita possíveis conflitos de escrita entre as várias *threads* que estão executando a avaliação simultaneamente.

Os dados são organizados em uma sequência específica e separados por ponto e vírgula (;). Essa sequência é a seguinte: (1) O *link* avaliado; (2) A porcentagem de acessibilidade do *link*; (3) O total de erros encontrados durante a avaliação; (4) O total de avisos identificados durante a avaliação; (5) Erros de Marcação; (6) Avisos de Marcação; (7) Erros de Comportamento; (8) Avisos de Comportamento; (9) Erros de Conteúdo e Informação; (10) Avisos de Conteúdo e Informação; (11) Erros de Apresentação e *Design*; (12) Avisos de Apresentação e *Design*; (13) Erros de Multimídia; (14) Avisos de Multimídia; (15) Erros de Formulários; (16) Avisos de Formulários.

Essa organização é fundamental para permitir uma fácil leitura e análise dos resultados da avaliação, possibilitando o uso posterior do arquivo para gerar um relatório completo e detalhado da acessibilidade do *site*, com informações específicas sobre cada *link* avaliado.

Após a conclusão da avaliação de todos os *links* processados, o navegador é liberado para uso posterior chamando a função `liberarNavegador()` que pode ser vista no Código 4.10. Ao final do processo de avaliação de todos os *links* processados, uma mensagem de conclusão é exibida no *console*: "Avaliação de todos os *links* concluída!".



## 5 Resultados

O presente estudo visa investigar a eficiência e o desempenho do *AccessCrawler*. Foram realizados diversos testes com o objetivo de avaliar o tempo de execução e a precisão dos resultados obtidos.

Para o *Web Crawler*, foram executados ciclos de coleta de *links*. No primeiro ciclo, o *Crawler* foi executado uma única vez para coletar *links* a partir de uma URL semente. No segundo ciclo, o *Crawler* foi executado a partir dos *links* encontrados no primeiro ciclo, ampliando o conjunto de *links* coletados e assim por diante. Os resultados foram comparados, analisando a quantidade total de *links* coletados em cada ciclo e a evolução do tamanho do conjunto de *links* ao longo dos ciclos.

### 5.1 *Web Crawler*

Os resultados obtidos revelam a atuação do *Web Crawler* na busca por *links* ao longo dos ciclos de execução. É evidente o aumento marcante na quantidade de *links* coletados a cada ciclo, destacando a habilidade do *Crawler* em explorar a vastidão da *Web* de maneira eficiente e sistemática. O sistema demonstrou eficácia em rastrear e seguir os *links*, resultando em uma expansão significativa do conjunto de *links* em cada fase subsequente do processo.

Entretanto, é interessante observar que, à medida que o *Crawler* avança nos ciclos de execução, os aumentos expressivos na quantidade de *links* coletados começam a diminuir. Essa desaceleração progressiva é um fenômeno natural, à medida que o *Crawler* se aproxima da saturação do espaço de busca disponível. Conforme os *links* são explorados e coletados, a capacidade de encontrar novos *links* diminui gradualmente, resultando em um crescimento mais moderado até eventualmente esgotar as oportunidades de descoberta.

No primeiro ciclo, o *AccessCrawler* foi iniciado com o *link* inicial fornecido <https://www.ufop.br/>. Ele navegou pela página de forma automatizada e sistemática, procurando os *links*. O tempo de execução foi de 1,4 segundos e resultou em 85 *links* encontrados. Não foram identificados *links* com erros durante o processo de verificação.

No segundo ciclo, o *Web Crawler* utilizou os *links* coletados no primeiro ciclo como sementes para iniciar a busca por novos *links*. O *Crawler* percorreu a *Web* a partir desses *links* e coletou um conjunto adicional de *links*. O tempo de execução foi de 7,2 segundos e resultou na coleta de 1.416 novos *links*. Durante o processo, não foram encontrados *links* com erros.

No terceiro ciclo, os *links* coletados no segundo ciclo foram utilizados como sementes

para iniciar a busca por novos *links*. O *Web Crawler* continuou percorrendo a *Web* a partir desses *links*, expandindo ainda mais o conjunto de *links* coletados. O tempo de execução foi de 57,6 segundos e resultou na coleta de 2.309 novos *links*. Nesse ciclo, foram identificados 846 *links* com erros e que não conseguiram ser processados.

No quarto ciclo, o *Web Crawler* utilizou os *links* coletados no terceiro ciclo como sementes para iniciar a busca por novos *links*. O *Crawler* continuou navegando pela *Web* e expandindo o conjunto de *links* coletados. O tempo de execução foi de 93,7 segundos e resultou na coleta de 2.500 novos *links*. Durante esse ciclo, foram encontrados 388 *links* com erros e que não conseguiram ser processados.

### 5.1.1 Análise do rastreo do AccessCrawler

A análise dos resultados do AccessCrawler revela uma visão detalhada de sua performance ao longo de vários ciclos de execução. A Tabela 1 a seguir apresenta os dados coletados durante o processo de busca por *links*, fornecendo *insights* sobre a quantidade de *links* processados, *links* encontrados, *links* com erro e o tempo de execução em cada ciclo.

Tabela 1 – Análise do processamento dos *links* pelo AccessCrawler

Ciclo	Processados	Válidos	Com Erro	Encontrados	Tempo (s)
1	1	1	0	85	1,4
2	85	85	0	1416	7,2
3	1416	570	846	2309	57,6
4	2309	1921	388	2500	93,7
5	2500	2153	347	2365	102,1
6	2365	1477	888	752	110,8
7	752	737	15	572	32,4
8	572	520	52	439	35,2
9	439	231	208	243	37,6
10	243	200	43	142	16,4
11	142	142	0	116	12,7
12	116	116	0	49	15,4
13	49	48	1	91	5,6
14	91	89	2	61	13,6
15	61	57	4	11	9,3
16	11	10	1	33	2,5
17	33	33	0	78	11,1
18	78	73	5	84	22,4
19	84	83	1	70	22,1
20	70	68	2	52	17,8
21	52	52	0	10	13,7
22	10	8	2	27	4,4
23	27	27	0	33	5,2
24	33	33	0	22	5
25	22	22	0	1	4,2
26	1	1	0	14	1,9
27	14	14	0	18	3,4
28	18	18	0	10	5
29	10	10	0	9	3
30	9	9	0	0	2,8
Total	11613	8808	2805	11612	675,5

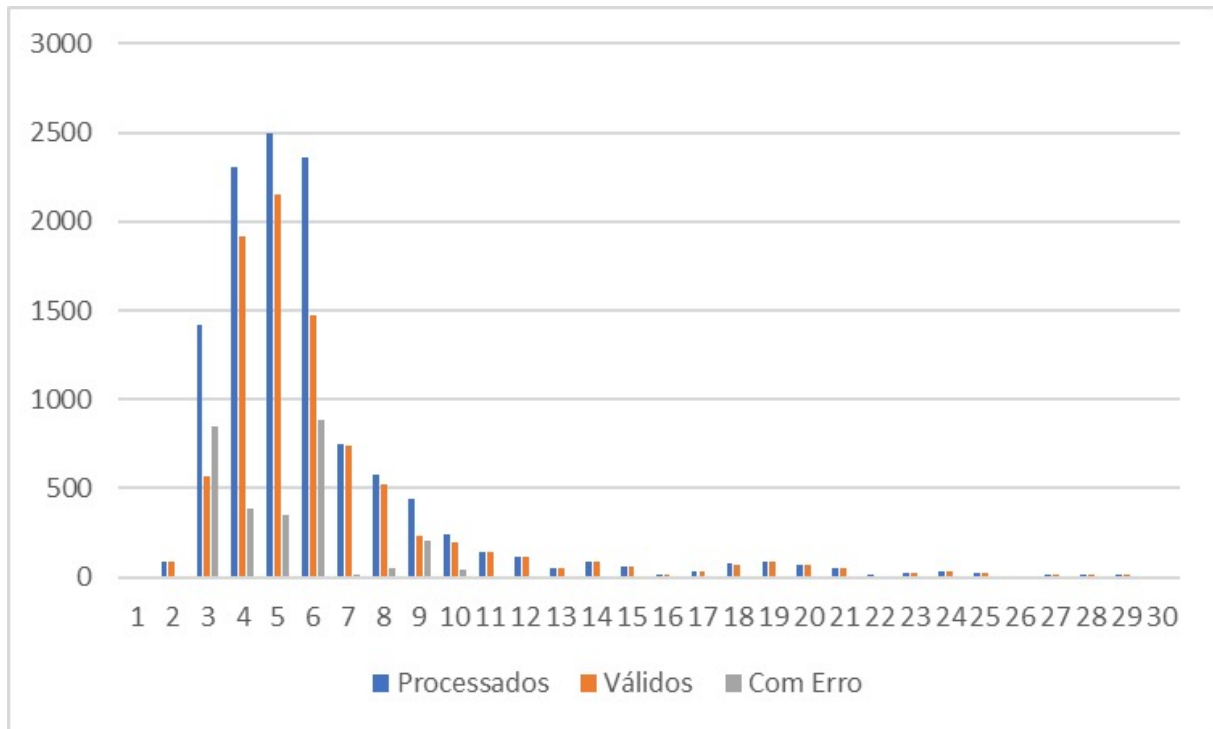
Fonte: Elaborado pelo autor, 2023

A Tabela reflete o progresso do *Crawler* à medida que explora a *Web* em busca de *links*. É perceptível a variação nos resultados, com um aumento inicial acentuado na quantidade de *links* coletados, seguido por uma desaceleração gradual. Isso indica o sucesso do *Crawler* em descobrir uma quantidade substancial de *links* em seus estágios iniciais, mas à medida que continua sua exploração, encontra cada vez menos *links* novos. Além disso, a presença de *links* com erros em determinados ciclos demonstra a complexidade e diversidade dos conteúdos *online*.

A Figura 9 ilustra a progressão do processamento dos *links* ao longo de vários ciclos. Nota-se uma diminuição na quantidade de *links* novos identificados à medida que

o processo avança, sugerindo que muitos desses *links* já foram previamente encontrados em outras páginas. Esse padrão sugere uma possível saturação na descoberta de *links* disponíveis no domínio da UFOP, indicando que o processo está se aproximando do final da sua exploração disponível.

Figura 9 – Desempenho de cada ciclo de processamento



Fonte: Elaborado pelo autor, 2023

Vale ressaltar que os números apresentados são baseados em um ambiente controlado e podem não refletir a totalidade dos *links* do domínio, especialmente considerando a aplicação dos filtros citados anteriormente na Seção 4.1.5. No entanto, essa análise oferece uma visão esclarecedora da dinâmica de busca e descoberta do *Web Crawler* em uma escala reduzida, e caso seja necessário analisar a totalidade, deve retirar os filtros de absurdos.

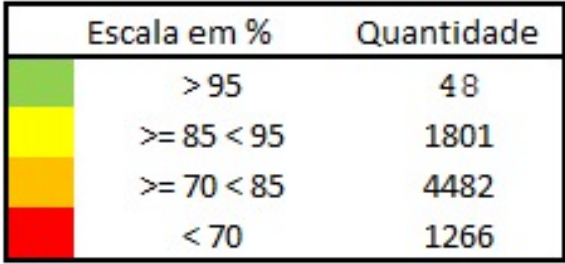
## 5.2 Análise dos resultados das avaliações de acessibilidade

Esta seção aborda a avaliação automática das páginas rastreadas pelo AcessCrawler. Durante a execução foram avaliados os 8808 *links* válidos que foram encontrados pelo *Crawler*. Utilizou-se um conjunto de 10 *threads* simultâneas para realizar os testes. O Avaliador de Acessibilidade foi configurado para processar um conjunto prévio de *links*, obtidos através do *Web Crawler*. O ASES analisou a acessibilidade dos *links*, medindo o tempo de execução e coletando métricas de acessibilidade, incluindo a porcentagem de acessibilidade, o total de erros e o total de avisos.

O AccessCrawler terminou as avaliações dos *links* válidos com 7 horas, o qual foram submetidos 8.808 *links*. Dentro dessa análise, 7607 *links* foram considerados como tendo sido avaliados de maneira bem-sucedida, enquanto 1201 apresentaram erros e não foram possíveis de serem avaliados. Dentre os *links* avaliados, se o que obteve a menor taxa de conformidade foi a página Educação Ambiental - ESCV <sup>1</sup>, atingindo 50,49%, e aquele com a maior conformidade encontrada foi a página do Colegiado de Ciência da Computação <sup>2</sup>, com uma taxa de 97,66%. A média de conformidade de todos os *links* analisados durante essas 7 horas foi de 78,6%. Em relação aos erros e avisos, a média registrada foi de 18 erros e 77 avisos.

Ao considerar os *links* avaliados, é importante mencionar que o ASES classificou um total de 48 *links* como de alto nível de conformidade, enquanto 1801 *links* obtiveram classificação de médio nível. Além disso, 4482 *links* foram categorizados como de baixo nível de conformidade e 1266 como de nível baixíssimo como podemos ver na Figura 10, isto revela que 75,66% das páginas do domínio ufop.br tem grau baixo ou baixíssimo de acessibilidade Web.

Figura 10 – Níveis de conformidade dos *links* avaliados



	Escala em %	Quantidade
	> 95	48
	>= 85 < 95	1801
	>= 70 < 85	4482
	< 70	1266

Fonte: Elaborado pelo autor, 2023

A Tabela 2 apresenta as médias de erros e avisos encontrados em cada seção do e-MAG, durante a avaliação dos *links*.

<sup>1</sup> Disponível em: <<http://www.cead.ufop.br/index.php/educacao-ambiental-escv?tmpl=component&print=1&page=>>. Acesso em: 20 de ago. 2023

<sup>2</sup> Disponível em: <<http://www.decom.ufop.br/cocic/manual-do-estudante/ensino-distancia/>>. Acesso em: 20 de ago. 2023

Tabela 2 – Média de Erros e Avisos

Seção	Erros	Avisos
Marcação	6	64
Comportamento	1	4
Conteúdo e Informação	8	4
Apresentação e <i>Design</i>	1	0
Multimídia	0	0
Formulários	0	2

A categoria de "Marcação" evidenciou um número considerável de avisos, enquanto a categoria de "Conteúdo e Informação" foi a que apresentou a maior média de erros. De maneira interessante, as seções e Comportamento de "Apresentação e *Design*" tiveram uma média de 1 erro e as seções "Multimídia" e "Formulários" não apresentaram erros. Isso pode indicar que as páginas já estão em conformidade com o e-MAG, ou que a grande maioria das páginas não possuem estes componentes.

Durante o período de 7 horas de avaliação, foi possível processar apenas uma pequena porcentagem dos *links* presentes no *site* da UFOP. Considerando que existem milhares de *links* em todo o *site*, a amostra avaliada representa apenas uma fração do total. Para obter uma média verdadeiramente representativa de conformidade, seria necessário um período de tempo muito maior para analisar os *links*.

Atualmente, após as 7 horas de avaliação, foram processados um total de 8.808 *links*. O tempo estimado para avaliar todos os *links* do domínio pode ser calculado pela fórmula:

$$TempoEstimado(horas) = \frac{7 \times TotalLinksDomínio}{8.808}$$

É importante destacar que essa estimativa é baseada nas condições atuais de execução do avaliador, e o tempo real pode variar de acordo com diferentes fatores, como a disponibilidade de recursos do sistema, a capacidade de processamento e o tamanho dos *links* a serem avaliados.

Pode-se utilizar a fórmula para ter uma previsão aproximada do tempo necessário para concluir a análise de todos os *links* da UFOP e obter uma média confiável da acessibilidade do *site*. Essa estimativa é uma ferramenta valiosa para planejar adequadamente o tempo e os recursos necessários para uma análise completa e abrangente da acessibilidade do *site* da universidade.

Além do tempo de execução do avaliador, os resultados das métricas de erros e avisos nas diferentes categorias forneceram *insights* valiosos sobre as áreas específicas que requerem maior atenção e correção. A categoria de "Marcação" foi a que apresentou o maior número de erros e avisos em todos os testes, indicando a necessidade de aprimoramentos

na estruturação do conteúdo. Por outro lado, as categorias de "Apresentação e *Design*", "Multimídia" e "Formulários" apresentaram o menor número de problemas, sugerindo uma boa conformidade nesse aspecto.

## 6 Conclusão

No escopo da Metodologia de Pesquisa DSR, este projeto abordou uma investigação profunda no desenvolvimento de um *Web Crawler* destinado a avaliar a acessibilidade das páginas no domínio da UFOP, onde os desafios inerentes à vastidão da *Web* e à diversidade das páginas eletrônicas se mostraram evidentes e como resultado, um artefato foi criado, o AccessCrawler.

Os objetivos delineados foram claros e todos cumpridos de maneira satisfatória. Desde o mapeamento e rastreamento dos *links* no domínio da UFOP até a avaliação da acessibilidade de cada página e a compilação dos resultados fornecidos pelo Avaliador de Acessibilidade, todas as metas foram atingidas de forma abrangente.

A busca pela resposta ao problema central, a viabilidade de criar uma estratégia para avaliar todas as páginas de um domínio, revelou-se positiva. Embora existam particularidades, como páginas com autenticação ou outras peculiaridades, a estratégia desenvolvida neste projeto, empregando um *Web Crawler* personalizado e um Avaliador de Acessibilidade, se mostrou eficaz para abordar essa questão.

A estratégia implementada para alcançar esses objetivos combinou o *Web Crawler*, capaz de coletar uma variedade significativa de *links*, com o Avaliador de Acessibilidade, que permitiu avaliar a conformidade de cada *link* e identificar erros e avisos. Esse casamento entre os dois componentes foi crucial para oferecer uma compreensão completa da acessibilidade de todas as páginas no domínio da UFOP.

Os resultados obtidos foram robustos e revelaram padrões interessantes. A evolução do processo de avaliação ao longo dos ciclos de execução do *Web Crawler* indicou que muitos dos *links* já haviam sido previamente descobertos.

Olhando para o futuro, diversos caminhos promissores se abrem para desenvolvimentos no projeto. Uma possível direção é a criação de uma interface *Web* para o AccessCrawler, o que proporcionaria uma maneira mais intuitiva e amigável de interagir com o programa. Além disso, uma expansão significativa do Avaliador de Acessibilidade poderia ser considerada, com a incorporação da capacidade de coletar e analisar uma gama mais ampla de dados sobre a avaliação dos *links*. Isso permitiria uma análise mais detalhada dos resultados e, possivelmente, a identificação de padrões mais complexos de conformidade e não conformidade. Ademais, a otimização das operações de verificação e processamento dos *links* também poderia ser alvo de melhoria, buscando tornar o processo ainda mais eficiente e rápido.



# Referências

- ASES. *Avaliador e Simulador de Acessibilidade em Sítios*. 2023. <<https://asesweb.governoeletronico.gov.br/>>. [Acessado em 3 de março de 2023]. Citado 3 vezes nas páginas 21, 22 e 38.
- BACH, C. F. et al. Diretrizes de acessibilidade: uma abordagem comparativa entre wcag e e-mag. *Revista Eletrônica de Sistemas de Informação*, v. 8, n. 1, 2009. Disponível em: <<http://www.periodicosibepes.org.br/index.php/reinfo/article/view/271/233>>. Citado na página 19.
- BAX, M. P. Design science: filosofia da pesquisa em ciência da informação e tecnologia. *Ciência da informação*, v. 42, n. 2, 2013. Citado na página 15.
- Brasil. *Decreto-Lei nº 5.296, de 2 de dezembro de 2004*. Presidência da República, 2004. Disponível em: <[https://www.planalto.gov.br/ccivil\\_03/\\_ato2004-2006/2004/decreto/d5296.htm](https://www.planalto.gov.br/ccivil_03/_ato2004-2006/2004/decreto/d5296.htm)>. Citado na página 14.
- BÜHLER, C. et al. Monitoring accessibility of governmental web sites in europe. In: SPRINGER. *Computers Helping People with Special Needs: 11th International Conference, ICCHP 2008, Linz, Austria, July 9-11, 2008. Proceedings 11*. [S.l.], 2008. p. 410–417. Citado 2 vezes nas páginas 16 e 26.
- CHEN, L.; WANG, D.; OLSEN, M. G. Towards distribution of web sites in a crawler used for large scale web accessibility assessment. CiteSeer. Citado na página 16.
- CTA - Centro Tecnológico de Acessibilidade. *Avaliação de Acessibilidade em Sites*. 2018. <<https://cta.ifrs.edu.br/avaliacao-de-acessibilidade-em-sites/>>. Citado 2 vezes nas páginas 20 e 22.
- Departamento de Governo Eletrônico. *eMAG - Modelo de Acessibilidade em Governo Eletrônico*. Brasil, 2014. Acesso em 16 de abril de 2023. Disponível em: <<https://emag.governoeletronico.gov.br>>. Citado na página 19.
- DHENAKARAN, S.; SAMBANTHAN, K. T. Web crawler - an overview. *International Journal of Computer Science and Communication*, v. 2, n. 1, p. 265–267, January-June 2011. Citado na página 23.
- Governo Federal do Brasil. *Ases - Avaliador e Simulador de Acessibilidade em Sítios*. Ministério do Planejamento, Orçamento e Gestão; Secretaria de Tecnologia da Informação; Departamento de Governo Eletrônico; Ministério da Educação; Secretaria de Educação Profissional e Tecnológica; Rede Nacional de Pesquisa e Inovação em Tecnologias Digitais; Projeto de Acessibilidade Virtual, 2016. Acesso em: 20 de março de 2023. Disponível em: <<https://softwarepublico.gov.br/social/ases>>. Citado na página 14.
- KAUSAR, M. A.; DHAKA, V.; SINGH, S. K. Web crawler: a review. *International Journal of Computer Applications*, Foundation of Computer Science, 244 5 th Avenue, #1526, New York, NY 10001 . . . , v. 63, n. 2, p. 31–36, 2013. Citado na página 23.

PARDINI, R. et al. Observatório da acessibilidade da web brasileira. In: DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO, UNIVERSIDADE FEDERAL DE MINAS GERAIS, BELO HORIZONTE, MG, BRASIL; CENTRO DE ESTUDOS SOBRE TECNOLOGIAS WEB - CEWEB.BR, NIC.BR / CGI.BR, São PAULO, SP, BRASIL. Minas Gerais, Brasil, 2021. Citado 3 vezes nas páginas 16, 24 e 25.

PIMENTEL, M.; FILIPPO, D.; SANTORO, F. M. Design science research: fazendo pesquisas científicas rigorosas atreladas ao desenvolvimento de artefatos computacionais projetados para a educação. *Metodologia de Pesquisa em Informática na Educação: Concepção da Pesquisa*. Porto Alegre: SBC, 2019. Citado 2 vezes nas páginas 15 e 17.

PLANEJAMENTO, O. e. G. Ministério do et al. *Apresentação do Sistema Ases Desktop: Metodologia de Desenvolvimento de Sistemas - Fase Visão*. 2016. Disponível em: <[https://softwarepublico.gov.br/social/articles/0003/8380/MDS\\_Fase\\_A01\\_-\\_Apresenta\\_\\_o\\_do\\_Sistema\\_1-0-3.pdf](https://softwarepublico.gov.br/social/articles/0003/8380/MDS_Fase_A01_-_Apresenta__o_do_Sistema_1-0-3.pdf)>. Citado na página 14.

Portal do Software Público. *Sobre o Software ASES*. 2017. Acesso em: 17 de abril de 2023. Disponível em: <<https://softwarepublico.gov.br/social/ases>>. Citado na página 20.

SANTOS, A. T. C. d. Uma análise da acessibilidade web dos sites das universidades federais do brasil com base no e-mag. 2023. Citado 2 vezes nas páginas 19 e 20.

SILVA, B. G. da; RODRIGUES, K. R. H. Accessibility challenges in web systems implementation. In: SBC. *Anais do IX Workshop sobre Aspectos da Interação Humano-Computador para a Web Social*. [S.l.], 2018. p. 105–116. Citado na página 14.

SLATIN, J. M.; RUSH, S. *Maximum accessibility: Making your web site more usable for everyone*. [S.l.]: Addison-Wesley Professional, 2003. Citado na página 19.

W3C. *Diretrizes de Acessibilidade para Conteúdo Web (WCAG) 2.1*. 2018. Recomendação W3C de 05 de Junho de 2018. [Online; accessed 15-April-2023]. Disponível em: <<https://www.w3c.br/traducoes/wcag/wcag21-pt-BR/#intro>>. Citado na página 19.

W3C. *Cartilha de Acessibilidade Web - Fascículo I*. s.d. <<https://www.w3c.br/pub/Materiais/PublicacoesW3C/cartilha-w3cbr-acessibilidade-web-fasciculo-I.html>>. [Acesso em: 02 mar. 2023]. Citado na página 14.

WAI. *Introduction to Web Accessibility*. 2005. Online. Disponível em: <<https://www.w3.org/WAI/fundamentals/accessibility-intro/>>. Citado na página 18.

Web Accessibility Initiative (WAI). *About W3C WAI*. 2023. Online. Disponível em: <<https://www.w3.org/WAI/about/>>. Citado na página 18.