PROPOSTA DE ARQUITETURA PARA UM SISTEMA DE DETECÇÃO DE PLÁGIO MULTI-ALGORITMO

Rodrigo Mesquita de Abreu

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Geraldo Bonorino Xexéo

Rio de Janeiro

Setembro de 2011

PROPOSTA DE ARQUITETURA PARA UM SISTEMA DE DETECÇÃO DE PLÁGIO MULTI-ALGORITMO

Rodrigo Mesquita de Abreu

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

_____

Prof. Geraldo Bonorino Xexéo, D.Sc.

_____

Prof. Geraldo Zimbrão da Silva, D Sc.

_____

Prof. Jorge Lopes de Souza Leão, Dr. Ing.

RIO DE JANEIRO, RJ – BRASIL

SETEMBRO DE 2011

Abreu, Rodrigo Mesquita de

Proposta de Arquitetura para um Sistema de Detecção de Plágio Multi-algortimo / Rodrigo Mesquita de Abreu – Rio de Janeiro: UFRJ/COPPE, 2011.

XIII, 92 p.: il.; 29,7 cm.

Orientador: Geraldo Bonorino Xexéo.

Dissertação (Mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2011.

Referências Bibliográficas: p. 89-92.

1. Extração de informação. 2. Detecção de plágio. 3.Similaridades entre documentos. I. Xexéo, Geraldo Bonorino II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

À minha família, minha amada e meus amigos, pelo carinho e incentivo.

Agradecimentos

Agradeço aos meus pais, Eliane e Rubens, pela educação, incentivo e carinho que me deram, bem como toda estrutura que me propiciaram para chegar até aqui.

Agradeço aos meus irmãos, Ricardo e Renan, por, apesar das pequenas brigas, sempre me compreenderem e me aliviarem de tarefas caseiras para que eu pudesse prosseguir meus estudos. Bem como por compreenderem minha ausência e estresse em momentos importantes de nossas vidas.

Agradeço a minha namorada, Kally, por todo apoio que me deu nesse último ano de mestrado, por ter aberto mão de seu carnaval e de suas férias para me fazer companhia na árdua tarefa de concluir um mestrado, por me compreender nos momentos em que não pude estar presente, bem como as festas que não pudemos curtir por eu estar atarefado.

Agradeço a minhas avós pela paciência durante meus estresses e minha ausência. Bem como, aos meus tios, tias, primos e primas pela compreensão.

Agradeço ao meu orientador, Xexéo, pela paciência e oportunidade. Agradeço também por um conselho que me deu quando eu ainda cursava minha graduação. Conselho esse que, embora eu não tenha seguido a risca, sem ele eu talvez não estivesse concluindo este mestrado hoje.

Agradeço ao professor Jano pelas oportunidades que me deu desde o final da minha a graduação, passando pela minha aceitação no mestrado e culminando no incentivo para que eu continuasse o mestrado mesmo após meu ingresso na Petrobras.

Agradeço aos professores Zimbrão e Leão por me concederem seu tempo e paciência, lendo esta dissertação e participando de minha banca. Espero que apreciem a leitura.

Agradeço a Fellipe Duarte pelo apoio durante meu último ano de mestrado, por suas dicas, pelo seu tempo e paciência, para me dar uma luz nos momentos de dificuldades na elaboração dessa dissertação.

Agradeço a professora Jonice Oliveira, que quando conheci ainda era aluna de doutorado do PESC, mas que, apesar do status, até hoje mantém sua simplicidade,

paciência e vontade de ajudar aos outros. Agradeço-a também pelas oportunidades me dadas no início desse mestrado.

Agradeço a André Korenchendler, Bruno Osiek, Carlos Eduardo Mello e Luis Fernando Orleans por estarem sempre dispostos a tirar minhas dúvidas durante o mestrado.

Agradeço aos alunos da turma de Busca e Recuperação da Informação de 2011/2 pela grande ajuda dada gerando dados para o experimento apresentado nesta dissertação. Espero que vocês tenham ainda mais sucesso do que eu estou tendo.

Agradeço aos amigos e colegas que fiz durante esse mestrado. Um dos grandes benefícios que esse mestrado me propiciou foi conhecer tantas pessoas legais e com tanta diversidade cultural.

Agradeço as pessoas que são responsáveis por manter o PESC funcionando, cuidando de alunos e professores e de toda a infraestrutura que necessitamos. Sendo assim, fica a minha lembrança ao pessoal da secretaria, do suporte, da segurança e da faxina. Obrigado pela paciência, conversas e sorrisos.

Agradeço aos meus amigos e colegas da Petrobras pela compreensão e apoio durante o período em que me dividi entre trabalho e estudos. Agradeço principalmente por terem me propiciado a flexibilidade que eu precisava para cumprir minha carga horária tanto no trabalho quanto no mestrado.

Por fim, agradeço a Deus, mas também aos Cientistas, que nos propiciem o mundo maravilhoso no qual vivemos.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

PROPOSTA DE ARQUITETURA PARA UM SISTEMA DE DETECÇÃO DE PLÁGIO MULTI-ALGORITMO

Rodrigo Mesquita de Abreu

Setembro/2011

Orientador: Geraldo Bonorino Xexéo

Programa: Engenharia de Sistemas e Computação

O plágio tem estado constantemente na mídia nos últimos anos. Com a popularização da Internet, é notável o aumento da disponibilidade de trabalhos prontos e a facilidade propiciada pelos meios digitais para cópia e manipulação de documentos e de seu conteúdo. Juntando-se a isso, tem-se a falta de informação e maturidade por parte dos estudantes, que tem a sua disposição tanta informação e tão pouco tempo para assimila-las, que por muitas vezes não compreendem as consequências do plágio, ou mesmo sabem como referenciar corretamente um trabalho. Esta dissertação tem por objetivo apresentar diferentes formas de geração e de detecção de plágio, e propor uma arquitetura para uma ferramenta de detecção de plágio multi-algoritmos, a ser utilizada em pesquisas futuras. A dissertação também apresenta uma validação para essa arquitetura, através da implementação de um protótipo para a mesma e execução de experimentos que demonstrem seus benefícios.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

PROPOSAL OF ARCHITECTURE FOR A MULTI-ALGORITHM PLAGIARISM DETECTION SYSTEM

Rodrigo Mesquita de Abreu

September/2011

Advisor: Geraldo Bonorino Xexéo

Department: Computer and System Engineering

Recently, plagiarism has constantly been in media. With the popularization of the Internet, the increasing availability of ready textual works is notable and the facilities provided by digital media to copy and manipulation of documents, and its contents,. In addition, there is the lack of information and maturity of the students, which have so much information in hand and short time to assimilate it. Students often do not know the plagiarism consequences or even know how to reference properly a work. This dissertation is intended to show different ways of generating and detecting plagiarism, and propose an architecture for a multi-algorithms plagiarism detection tool, that will be used in future researches. The work also presents a validation for the architecture, by implementing a prototype and executing experiments that demonstrate its benefits.

Sumário

Índice de Figuras

Índice de Tabelas

1 Introdução

1.1 Motivação

A expansão da Internet, tanto em número de usuários quanto na diversidade de meios de representação e disponibilização da informação, permite que cada vez mais conteúdo intelectual esteja acessível àqueles que necessitam de uma base bibliográfica para apoiar seus trabalhos. Entretanto, essa expansão se deu de forma rápida, se comparada com a velocidade com que conhecimentos sobre ética e propriedade intelectual são disseminados e assimilados pelos novos e antigos usuários.

O plágio tornou-se uma constante não só na Internet como no meio acadêmico, tornando difícil a distribuição de méritos e o incentivo àqueles que realmente trabalham na evolução de uma linha de pesquisa.

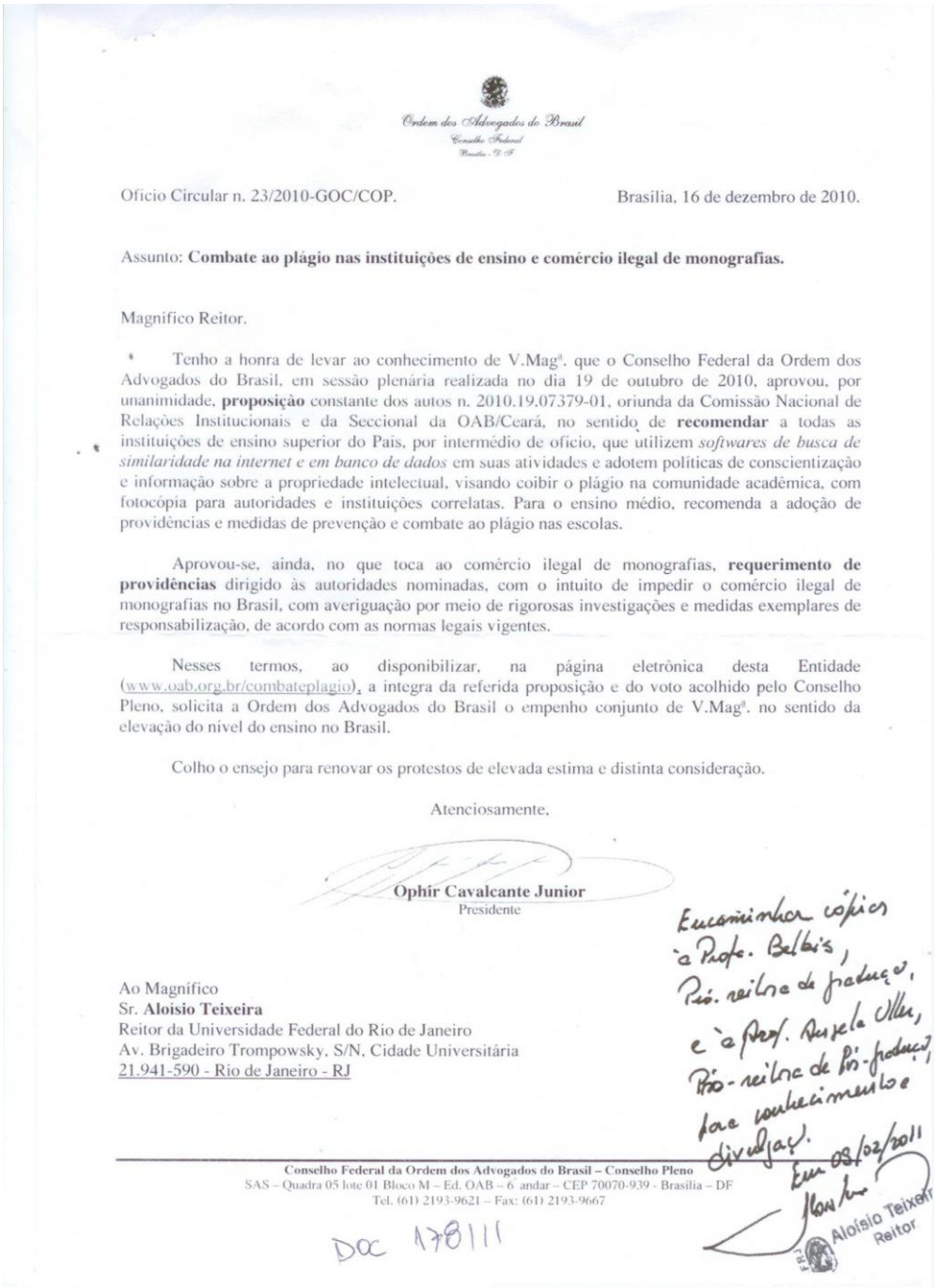
Segundo pesquisa realizada por Silva e Domingues (2008), entre alunos de Pós-Graduação *Lato sensu*, os estudantes tem consciência de que plágio constitui crime**.** Porém, o conceito de plágio parece não ter sido bem assimilado. Silva e Domingues (2008) concluem:

"...nas questões que dizem respeito ao conceito de plágio e à forma correta de utilização de conteúdos produzidos por outros autores, os alunos demonstram despreparo e falta de conhecimento."

Segundo Maurer et al (2006) "o plágio é considerado o mais sério problema de conduta escolar e a academia em todo mundo está fazendo esforços para educar alunos e professores". Maurer et al (2006) também apresenta o resultado de uma pesquisa realizada em 2005 que mostra que 40% dos estudantes admitem fazer uso de plágio, e ao comparar com outra pesquisa realizada em 1999, deixa evidente um crescimento de 300%.

No Brasil, os esforços no combate ao plágio também podem ser percebidos. Para exemplificar esse esforço, pode-se citar um ofício enviado pelo então presidente do Conselho Federal da Ordem dos Advogados do Brasil, Sr. Ophir Cavalcante Junior, em dezembro de 2010, ao então reitor da Universidade Federal do Rio de Janeiro, Sr. Aloísio Teixeira. O ofício em questão, reproduzido na Figura 1, informa que o Conselho

aprovou em sessão plenária, por unanimidade, a proposição de recomendar a **todas as instituições de ensino superior do país** que "utilizem *softwares de busca de similaridade na internet e em banco de dados* em suas atividades e adotem políticas de conscientização e informação sobre a propriedade intelectual, visando coibir o plágio na comunidade acadêmica". Já para o ensino médio, o Conselho "recomenda a adoção de providências e medidas de prevenção e combate ao plágio nas escolas".



Ordem dos Advogados do Brasil
Conselho Federal
Brasília - D.F.

Ofício Circular n. 23/2010-GOC/COP. Brasília, 16 de dezembro de 2010.

Assunto: **Combate ao plágio nas instituições de ensino e comércio ilegal de monografias.**

Magnífico Reitor.

Tenho a honra de levar ao conhecimento de V.Magª. que o Conselho Federal da Ordem dos Advogados do Brasil, em sessão plenária realizada no dia 19 de outubro de 2010, aprovou, por unanimidade, **proposição** constante dos autos n. 2010.19.07379-01, oriunda da Comissão Nacional de Relações Institucionais e da Seccional da OAB/Ceará, no sentido de **recomendar** a todas as instituições de ensino superior do País, por intermédio de ofício, que utilizem *softwares de busca de similaridade na internet e em banco de dados* em suas atividades e adotem políticas de conscientização e informação sobre a propriedade intelectual, visando coibir o plágio na comunidade acadêmica, com fotocópia para autoridades e instituições correlatas. Para o ensino médio, recomenda a adoção de providências e medidas de prevenção e combate ao plágio nas escolas.

Aprovou-se, ainda, no que toca ao comércio ilegal de monografias, **requerimento de providências** dirigido às autoridades nominadas, com o intuito de impedir o comércio ilegal de monografias no Brasil, com averiguação por meio de rigorosas investigações e medidas exemplares de responsabilização, de acordo com as normas legais vigentes.

Nesses termos, ao disponibilizar, na página eletrônica desta Entidade (www.oab.org.br/combateplagio), a íntegra da referida proposição e do voto acolhido pelo Conselho Pleno, solicita a Ordem dos Advogados do Brasil o empenho conjunto de V.Magª. no sentido da elevação do nível do ensino no Brasil.

Colho o ensejo para renovar os protestos de elevada estima e distinta consideração.

Atenciosamente,

Ophir Cavalcante Junior
Presidente

Encaminhar cópias à Profa. Belkis, Pró-reitora de graduação, e à Profa. Rosiska Oliveira, Pró-reitora de Pós-graduação, para conhecimento e divulgação. Em 08/02/2011

Ao Magnífico
Sr. **Aloísio Teixeira**
Reitor da Universidade Federal do Rio de Janeiro
Av. Brigadeiro Trompowsky, S/N, Cidade Universitária
21.941-590 - Rio de Janeiro - RJ

Aloísio Teixeira
Reitor

Conselho Federal da Ordem dos Advogados do Brasil – Conselho Pleno
SAS – Quadra 05 lote 01 Bloco M – Ed. OAB – 6 andar – CEP 70070-939 - Brasília – DF
Tel. (61) 2193-9621 – Fax: (61) 2193-9667

Doc 178111

C

Figura 1 - Ofício enviado ao Magnífico Reitor da UFRJ pela OAB recomendando o uso de software que auxilie no combate ao plágio em todas as instituições de ensino superior do Brasil.

1.2 Objetivo do trabalho

É visando atacar o problema do plágio que este trabalho tem como objetivo desenvolver uma arquitetura para uma ferramenta que auxilie na detecção e prevenção do plágio textual em meio acadêmico, com foco em monografias, dissertações e teses, e que seja flexível o suficiente para que sirva de plataforma para implementação, teste e execução de novas técnicas de detecção de plágio, para que seja possível verificar a eficácia de novos e antigos algoritmos computacionais e de suas combinações.

Esta dissertação propõe uma arquitetura que divide o problema em cinco etapas: a extração de consultas para busca por possíveis fontes, a recuperação de documentos a partir das consultas, a detecção de similaridades entre o documento analisado e os documentos recuperados, o refinamento das comparações através da reestruturação dos documentos e a formulação de índices de plágio que melhor representem os resultados. Cada etapa por si só pode dar origem a diferentes trabalhos, de proporções até maiores do que as pretendidas nesta dissertação, porém a intenção deste trabalho é de encontrar um caminho para resolução do problema, implementando uma solução para detecção de plágio com índices de desempenho satisfatórios e abrindo caminho para trabalhos que aperfeiçoem ainda mais as técnicas utilizadas em cada etapa.

1.3 Divisão dos capítulos

Esta dissertação está dividida em sete capítulos. O primeiro capítulo contém esta introdução, composta da motivação, do objetivo e da divisão do trabalho. O segundo capítulo apresenta uma revisão do tema principal do trabalho, plágio, conceito, como é feito e de que forma é detectado. O terceiro capítulo introduz diferentes técnicas de processamento de texto que podem auxiliar, de alguma forma, à detecção de plágio.

O quarto capítulo desta dissertação apresenta a arquitetura proposta, detalhando suas etapas e aplicando sobre ela um teste de mesa. Já o quinto capítulo apresenta a validação da proposta, na qual detalha a implementação do protótipo TEXPLAN (Textual Plagiarism Analyzer), apresentando suas especificações, limitações e benefícios, e o experimento que demonstra seu funcionamento e efetividade.

O capítulo seis descreve as conclusões obtidas do trabalho, conclusões quanto aos algoritmos e técnicas utilizadas, resultados obtidos durante o desenvolvimento, como a

detecção de plágio real em monografia de conclusão de graduação, além dos trabalhos futuros. Por fim, o capítulo sete apresenta a bibliografia utilizada.

2 Plágio

Muitas são as definições atribuídas a plágio. A Enciclopédia Britânica (ENCYCLOPÆDIA BRITANNICA INC., 1984) define plagiar como "ato de falar dos escritos de outra pessoa e fazê-los passar como próprios." Já a *Modern Language Association* adota a definição de Gibaldi (2003), onde plágio é o uso de ideias ou expressões de outra pessoa, sem reconhecer a fonte, e plagiar é dar a impressão de que você escreveu ou pensou algo que na verdade foi obtido de outra pessoa.

A definição dada por Gibaldi (2003), apresentada anteriormente, em que o uso de ideias alheias sem as devidas referências é caracterizado como plágio, amplia o conceito de plágio do simples "copiar e colar" parte de uma obra para um conceito que abrange a extração de ideias de trabalho alheio e a reescrita das mesmas em outro trabalho. Esse conceito é importante neste trabalho, pois sua proposta é analisar documentos suspeitos de plágio e encontrar não somente trechos integralmente copiados, mas também trechos ou ideias parafraseadas.

O plágio pode ser aplicado nas mais diversas áreas, em diferentes formas. Este trabalho foca no plágio literário. O plágio literário é complexo e pode abranger obras de difícil avaliação como livros de história. Além disso, em determinadas épocas escolares, como no ensino fundamental, alguns níveis de plágio podem não ser considerados tão graves quanto se fossem aplicados em trabalhos no nível de ensino superior. Com base nessa grande abrangência e complexidade do plágio literário que foi necessário dar um foco a este trabalho, sendo ele então focado no plágio literário em monografias, dissertações e teses.

2.1 Formas de plágio

Cada autor tem sua forma de classificar o plágio, sendo as classificações apresentadas em seus trabalhos aquelas que foram consideradas por eles como as mais adequadas ao prosseguimento de seus trabalhos. Nesta seção são apresentadas diferentes formas de classificação e, ao final da mesma, as formas são compiladas em uma única taxonomia para classificação de plágio.

Leung & Chan (2007) dividem as técnicas em três casos:

Cópia de fonte sem versão eletrônica – neste caso apenas seres humanos são capazes de analisar as possíveis fontes de plágio;

Cópia de fonte com versão digital – simples cópia do texto original, podendo ser detectada através do uso de ferramentas de busca na Web ou por sistemas de detecção de plágio que tenham indexado a fonte;

Cópia de fonte com versão digital e modificação do conteúdo – podem ser detectadas por sistemas de detecção de plágio, porém com maior dificuldade devido às alterações. Segundo Leung & Chan (2007), as modificações mais comuns são:

- **Mudança de Voz**: A frase tem a voz verbal alterada. Ex.: Maria comeu o queijo. -> O queijo foi comido por Maria.
- **Mudança de Tempo**: O tempo verbal da frase é alterado. Ex.: Maria come o queijo. -> Maria comeu o queijo.
- **Uso de Sinônimos**: Uma ou mais palavras da frase são substituídas por sinônimos. Ex.: Maria comeu o queijo. -> Maria devorou o queijo.

Segundo Weber-Wulff (2010), Weber-Wulff e Wohnsdorf foram capazes de diferenciar seis formas de plágio, conforme descritas abaixo:

Copy & Paste – simples cópia do texto original ou de parte dele;

Translations – simples tradução do texto de sua língua original para língua utilizada no novo documento, podendo ser feita manualmente ou com o uso de ferramentas de tradução automática;

Shake & Paste Collections – trata-se da coleta de trechos de diversas fontes e posterior ordenação aleatória dos mesmos no novo documento, sendo facilmente identificada através de leitura humana devido às freqüentes mudanças de estilo, escolha de palavras e formatação;

Clause Quilts ou Mosaics – nesta forma trechos de textos de diferentes fontes são coletados e alterados. As alterações são feitas utilizando *patchwriting*, termo definido por Howard (1992) como "copiar um texto fonte e então apagar algumas palavras, alterar estruturas gramaticais e/ou substituir algumas palavras por seus sinônimos" (tradução livre);

Structural Plagiarism – ou Plágio de Estrutura (tradução livre). Nesta forma o texto original é parafraseado utilizando a mesma estrutura argumentativa, fontes, experimentos e, até mesmo, resultados;

Collusion – nesta forma um grupo de autores acorda em escrever um único documento, então, cada um faz suas modificações no documento original e se declara como único autor. Esse tipo de plágio pode ser facilmente detectado visto que os autores terão algum vínculo, como, por exemplo, fazer parte de uma classe de graduação, o que torna o universo de documentos, a serem comparados, menor do que o de uma comparação com toda a comunidade acadêmica.

A Tabela 1 mostra as relações entre as classificações apresentadas por Chan & Leung e Weber-Wuff & Wohnsdorf. É possível ver que Weber-Wuff & Wohndorf ignoram o plágio de fonte não digital. No entanto, é importante manter a classificação "cópia de fonte sem versão eletrônica" (LEUNG e CHAN, 2007), isso porque, apesar de não se poder acusar um documento de plágio sem se ter o documento fonte em mãos, é possível avaliar um documento isoladamente e aumentar o nível de suspeita sobre ele com o uso de *intrinsic plagiarism analysis* (STEIN, LIPKA e PRETTENHOFER, 2010) ou análise intrínseca de plágio (tradução livre), conforme será discutido mais adiante. Também é interessante observar que as classificações de Weber-Wulff & Wohnsdorf complementam a classificação "cópia de fonte com versão digital e modificação do conteúdo" (LEUNG e CHAN, 2007), que originalmente citavam apenas alterações gramaticais, que por sua vez podem ser definidas como subconjunto da classificação "Clause Quilts ou Mosaics".

Tabela 1 - Comparação das classificações de Leung & Chan e Weber-Wulff & Wohnsdorf.

Leung & Chan	Weber-Wulff & Wohnsdorf
Cópia de fonte sem versão eletrônica	
Cópia de fonte com versão digital	Copy & Paste
Cópia de fonte com versão digital e modificação do conteúdo	Translations
	Shake & Paste Collections
	Clause Quilts ou Mosaics
	Structural Plagiarism
	Collusion

Abaixo são apresentadas duas formas de classificação (Figura 2 e Figura 3) definidas neste trabalho tomando como base as classificações anteriormente apresentadas. Essas duas formas de classificação originaram as operações de criação de plágio descritas mais adiante.

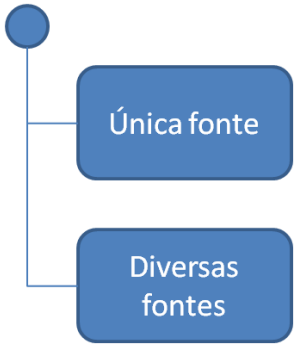


Figura 2 - Classificação quanto à multiplicidade de fontes.

A primeira classificação, apresentada na Figura 2, diz respeito ao número de fontes utilizadas em um texto plagiado. Um plágio pode ter uma única fonte ou ser uma composição de diversas fontes. Valendo lembrar que a identificação de diferentes fontes pode ser dificultada através de métodos como o *Shake & Paste Collections*, descrito por Weber-Wulff e Wohnsdorf (WEBER-WULFF, 2010).

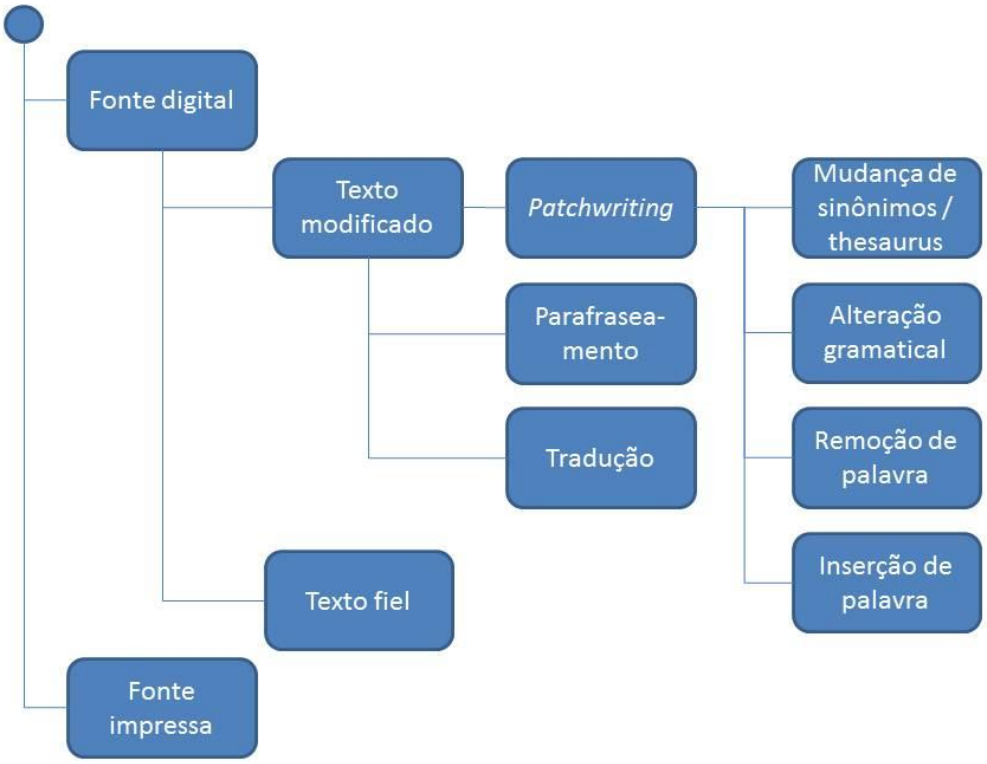


Figura 3 - Classificação das técnicas de plágio.

A Figura 3 apresenta a classificação das técnicas de plágio aplicadas sobre um texto. O plágio pode consistir na aplicação consecutiva de várias dessas técnicas. Por exemplo, um texto pode ser inicialmente obtido de uma fonte impressa, digitalizado, traduzido e então sofrer alterações gramaticais. Quanto maior o número de técnicas aplicadas, maior será a dificuldade de se obter o texto original ou até mesmo de levantar suspeitas quanto à originalidade do trabalho.

Também é possível classificar as formas de detecção de plágio como intra-corpus ou extra-corpus (LANCASTER e CULWIN, 2005), onde plágio intra-corpus é feito utilizando-se de um conjunto restrito de documentos, como por exemplo, plágio de trabalho entre colegas de um mesmo curso, e extra-corpus é caracterizado pelo plágio de documentos externos, como documentos provenientes da Internet. A Figura 4 ilustra essa taxonomia.

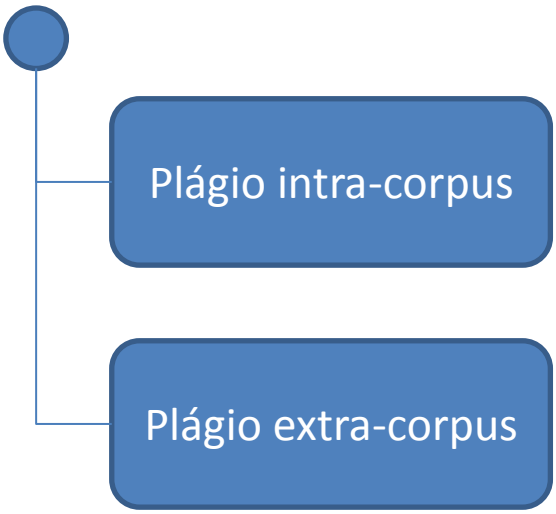


Figura 4 - Classificação quanto a origem da(s) fonte(s) do plágio

2.2 Formas de detecção de plágio

Noynaert (2006) apresenta três métodos para detecção de plágio:

Método de perguntas - Consiste em questionar o autor quanto ao seu texto. Pode ser suportado por softwares como o *Glatt Plagiarism Screening System*, mencionado por Noynaert (2006), que retira palavras do texto e solicita que o aluno preencha as lacunas;

Método de estilo de escrita - Verifica se o estilo de escrita se mantém durante todo o texto e se o texto se assemelha a textos anteriormente escritos pelo suposto autor;

Método de comparação - Compara o texto suspeito com outros textos em uma base de documentos ou na Internet.

O método de perguntas é interessante tanto para uso por professores que queiram argüir seus alunos em sala de aula, quanto para autores que queiram auto-avaliar seus textos e verificar se seu estilo de escrita está consistente, porém dificilmente se adequaria na avaliação de artigos submetidos a revistas, por exemplo. Dessa forma, esta dissertação levará em consideração apenas os outros dois métodos.

Tripolitis (2002) lista alguns fatores que podem denunciar um plágio:

- **Mudanças de vocabulário** – vocabulário amplo pode indicar mais de um autor;
- **Incoerência** – texto aparenta ser escrito por mais de uma pessoa;
- **Pontuação** – com o uso do mesmo esquema de pontuação dois textos passam a ter sua forma muito parecida.

- **Dependência de certas palavras ou frases** – uso de palavras ou frases comuns a outro autor;
- **Excesso de similaridade entre textos**;
- **Sequências longas de caracteres ou palavras em comum**;
- **Ordem de semelhanças** – palavras ou frases em comum estão na mesma ordem nos textos;
- **Mesma frequência de palavras em diferentes textos**.

A Tabela 2 apresenta uma relação entre os estudos de Noynaert (2006) e Tripolitis (2002). Além das técnicas descritas acima, também é possível acrescentar à tabela outras técnicas de detecção de plágio que serão descritas na seção 2.3.

Tabela 2 - Relação entre os estudos de Noynaert (2006) e Tripolitis (2002).

Noynaert	Tripolitis
Método de perguntas	
Método de estilo de escrita	Mudança de vocabulário
	Incoerência
Método de comparação	Pontuação
	Dependência de certas palavras ou frases
	Excesso de similaridade entre textos
	Sequências longas de caracteres ou palavras em comum
	Ordem de semelhanças
	Mesma frequência de palavras em diferentes textos

2.3 Métodos

Os algoritmos utilizados pelos Sistemas de Detecção de Plágio existentes são tratados como segredos industriais (WEBER-WULFF, 2010) (LUKASHENKO, GRAUDINA e GRUNDSPENKIS, 2007), o que dificulta a identificação dos principais algoritmos utilizados no mercado.

Embora não seja possível comprovar quais métodos são efetivamente utilizados nos sistemas disponíveis hoje no mercado, esta seção faz um levantamento dos métodos de detecção de plágio disponíveis na literatura. Os métodos são apresentados em duas etapas, primeiramente são apresentados os métodos de estilo de escrita e posteriormente os métodos de comparação.

2.3.1 Métodos de estilo de escrita

A análise intrínseca de plágio (STEIN, LIPKA e PRETTENHOFER, 2010) utiliza-se da identificação do estilo de escrita aplicado a um documento para determinar quais fragmentos desse documento se destacam dos demais e assim sinalizá-los como possíveis plágios. Os métodos utilizados nesse tipo de análise não necessariamente farão uso de um corpus e consequentemente demandam menor tempo computacional. Um corpus poderá ser útil para apontar que autores têm estilo de escrita similar ao do documento analisado ou mesmo se o estilo do documento foge ao estilo utilizado pelo mesmo autor em outros documentos.

O tipo de análise apresentado por Stein et al (2010) divide o documento suspeito em seções e aplica estilometria para identificar quais seções fogem do padrão do documento. As seções que se destacam são chamadas de *outliers*.

2.3.2 Métodos de comparação

Os métodos mais comumente utilizados na detecção de plágio através da comparação entre documentos envolvem radicalização ou *fingerprinting* (MAURER, KRAPPE e ZAKA, 2006), onde radicalização consiste em reduzir as palavras do documento ao seu radical.

Fingerprinting é uma abordagem para a comparação de documentos, inicialmente apresentada por Manber (1994), que assume que as representações compactas (*fingerprint*) de documentos similares serão similares, enquanto as representações compactas de documentos não similares serão diferentes, ambas as afirmações com alta probabilidade. Essa abordagem reduz a comparação de documentos à comparação de suas *fingerprints*.

Vale ressaltar que existem métodos preventivos que podem facilitar a detecção de plágio. Esses métodos consistem em anexar ao documento metadados que, no processo de cópia para um novo documento, deverão ser carregados sem que o autor do plágio tenha conhecimento. Este trabalho não dará mais detalhes sobre esses métodos visto que os documentos a serem analisados como possíveis fontes de plágio não se limitarão a documentos que possuam esse tipo de metadado.

Outra maneira de classificar métricas de plágio é em métodos semânticos ou métodos estatísticos (LUKASHENKO, GRAUDINA e GRUNDSPENKIS, 2007). Por sua vez,

os métodos estatísticos são classificados por Gruner e Naven como independentes de linguagem ou sensíveis a linguagem (LUKASHENKO, GRAUDINA e GRUNDSPENKIS, 2007).

Os métodos de comparação tentam identificar documentos cujo nível de similaridade passa do estágio de aceitável para plágio, porém o nível aceitável de similaridade é relativo. Lancaster & Culwin (2004) lembram que os níveis de similaridade aceitáveis podem variar de acordo com o tema ou classe de pessoas, sendo insuficiente adotar um valor máximo para eles.

Os métodos de comparação enfrentam um problema relativo ao tamanho do corpus a ser utilizado na comparação. O tempo de análise será diretamente proporcional a quantidade de documentos fonte comparados ao documento suspeito. Embora no caso de uma análise intra-corpus o corpus tenha um tamanho finito determinado, esse tamanho ainda pode tornar o tempo de processamento inviável para uma análise de todos os documentos, e o mesmo ocorrerá quando se tratar de uma análise extra-corpus, cujo tamanho será o tamanho da Web. Uma maneira de se evitar esse problema é limitando o tamanho do corpus analisado através de filtros, o que leva ao problema de aperfeiçoar a seleção de documentos de modo que esta leve a um melhor resultado na detecção de plágio.

2.3.2.1 Métrica de Pares de Palavras

Lancaster & Culwin (2004) defendem que as métricas utilizadas na detecção de similaridade entre pares de documentos devem obedecer a três critérios para serem utilizados: Efetividade, onde a métrica deve retornar os mais altos valores para documentos similares e valores baixos para documentos distintos, Eficiência Computacional e Eficiência ao Tutor, onde a métrica deve dar o mínimo de trabalho ao Tutor, que é o ser humano responsável por operar o PDS e dar a palavra final sobre a autenticidade do documento. Baseados nesses três critérios, Lancaster & Culwin (2004) defendem que a Métrica de Pares de Palavras seria a mais eficiente para detecção de plágio.

A Métrica de Pares de Palavras (LANCASTER e CULWIN, 2004) consiste em um método que gera todos os pares de palavras consecutivas encontradas em um documento e as compara com os pares de palavras de outro documento. A similaridade entre esses dois documentos terá um valor entre 0 e 100 encontrado pela seguinte equação:

$$\frac{100 * (c1 + c2)}{(c1 + c2) + (u1 + u2)},$$

onde c1 e c2 são as quantidades de pares de palavras comuns no primeiro e no segundo documento, respectivamente, e u1 e u2 representam a quantidade de pares de palavras únicas no primeiro e no segundo documento, respectivamente.

2.3.2.2 N-gramas

Barrón-Cedeño & Rosso (2009) apresentaram o método para detecção de plágio utilizando N-gramas para comparação de documentos descrita abaixo:

1. O documento suspeito é dividido em sentenças (s_i);
2. Cada sentença s_i é dividida em n-gramas de palavras;
3. Um documento *d* é completamente dividido em n-gramas de palavras;
4. Cada sentença s_i é buscada no documento *d*, comparando-se os n-gramas.

A similaridade entre uma sentença s_i e um documento *d* é calculada por Barrón-Cedeño & Rosso (2009) através da seguinte fórmula apresentada por Lyon et al:

$$C(si \mid d) = \frac{|N(si) \cap N(d)|}{|N(si)|},$$

onde *N(x)* representa o conjunto de n-gramas de palavras de um texto *x*. Após executar o procedimento para todos os documentos *d* originais, se o valor máximo encontrado para $C(s_i|d)$ for maior que um valor mínimo pré-definido, o documento analisado será indicado pelo método como possível plágio.

2.3.2.3 Métodos de redução de corpus

A redução do corpus, a ser analisado por um método de comparação, acarreta em ganhos computacionais, pois reduz o número de comparações a serem efetuadas, e aumento de precisão, pois ao reduzir o corpus, tende-se a manter no mesmo os documentos com maior probabilidade de satisfação do propósito da análise, no caso, os documentos com maior probabilidade de serem fontes de plágio de um documento suspeito *d*.

É possível encontrar na literatura métodos que reduzam o corpus após uma análise prévia de todos os documentos que o compõem, como é o caso do método apresentado por Benedí et al (2009) que utiliza distância de Kullback-Leibler para filtrar os

documentos mais próximos ao documento suspeito. Porém esses métodos não se aplicam quando o corpus é tão grande quanto a Web, por exemplo.

2.3.2.4 Métodos de recuperação de corpus

Para examinar ambientes grandes como a Web, estão disponíveis ferramentas de busca como o Google[1]. O Google é uma ferramenta muito poderosa, capaz de retornar resultados em menos de um segundo, utilizando diversos algoritmos que aumentam sua eficácia em retornar os resultados desejados.

Este trabalho não utiliza um corpus fixo, mas o constrói automaticamente a partir de resultados de buscas por documentos. Buscas que podem ser feitas em diferentes bases de documentos, incluindo a Web com o uso de ferramentas como o Google. Logo, os resultados deste trabalho dependem da escolha das consultas e das bases de documentos utilizadas. Para aumentar a eficácia das buscas feitas, o capítulo 3 deste trabalho propõe alguns métodos para definição das buscas a serem executadas.

2.4 Ferramentas

Ao término da revisão bibliográfica feita sobre o assunto plágio, percebe-se que o foco da maioria dos autores e das informações sobre as ferramentas disponíveis está em descobrir similaridades entre documentos. A maioria das ferramentas tenta identificar quão similares são dois documentos, ou dois trechos de documentos, com o menor custo de tempo possível, devido ao grande número de documentos disponíveis para comparação. Os resultados retornados são, em sua maioria, dois trechos de documentos e sua medida de similaridade. As ferramentas consideram os trechos retornados como suspeitos, mas nunca dão certeza sobre ser ou não um plágio. Cabe à ferramenta disponibilizar um relatório de similaridades que facilite a análise humana e a um analista humano que diga se o texto é um plágio ou não. Nos próximos parágrafos serão apresentadas algumas das ferramentas disponíveis, que exemplificam o que foi afirmado neste parágrafo e o estado atual de desenvolvimento da área.

2.4.1 Plagiarism.org [2]

Plagiarism.org disponibiliza diversas ferramentas que ajudam tanto na prevenção, quanto na identificação do plágio. Entre as ferramentas podemos destacar a Turnitin[3],

[1] Disponível em http://www.google.com
[2] Disponível em http://www.plagiarism.org
[3] Disponível em http://www.turnitin.com

"líder mundial em prevenção de plágio acadêmico e checagem de originalidade" (IPARADIGMS, 2011a). Além do Turnitin, também estão disponíveis o iThenticate, para prevenção de plágio em ambiente corporativo, o WriteCheck, que é uma ferramenta para auxilia o aluno indicando a necessidade de referência, e o Turnitin for Admissions, que auxilia na prevenção de plágio em pedidos de admissão a cursos de graduação detectando plágio, reutilização de pedidos anteriores, respostas duplicadas, documentos comprados, entre outros problemas de similaridade (IPARADIGMS, 2011b).

Todas as ferramentas em Plagiarism.org são pagas e estão disponíveis para alunos, professores e para ambientes empresariais.

O funcionamento das ferramentas é descrito em (IPARADIGMS, 2011a) da seguinte forma:

1. A ferramenta cria um *fingerprint* do documento submetido através de algoritmos não revelados;
2. O *fingerprint* criado é cruzado com centenas de milhares de documentos presentes em uma base própria do sistema;
3. Em paralelo, *web crawlers* são utilizados para vasculhar a internet em busca de possíveis fontes de plágio;
4. Um relatório colorido indicando possíveis fontes de plágio e URL's para os mesmos é gerado.

2.4.2 COPS

Copy Protection System (COPS) faz parte do Stanford Digital Library System[4]. COPS é um detector de cópias para bibliotecas digitais, verificando se um novo documento é similar de maneira suspeita de um ou mais documentos já existentes na biblioteca.

Clough (2000) explica que no COPS "os documentos são separados em sentenças e reagrupados em sequências de sentenças chamadas de *chunks*". Clough (2000) continua e explica que as sentenças são armazenadas utilizando um algoritmo de hash e posteriormente utilizadas para comparação com *chunks* de outros documentos registrados na biblioteca digital. "Caso os documentos compartilhem de um número

[4] Disponível em http:// www-diglib.stanford.edu

pré-definido de sentenças, uma violação é informada e um humano deve verificar a violação determinar qual o problema".

2.4.3 SCAM

O Stanford Copy Analysis Mechanism (SCAM) também faz parte do Stanford Digital Library System, tendo sido desenvolvido a partir da experiência obtida com o COPS (CLOUGH, 2000).

Segundo Clough (2000), a principal diferença entre SCAM e o COPS está no fato do primeiro ter palavras como a unidade básica de comparação, enquanto o segundo constrói sua tabela hash a partir de sentenças. Ainda segundo Clough (2000), SCAM agrupa palavras em conjuntos, chamados de *chunks*, e utiliza um derivado do Vector-Space Model para encontrar similaridades entre os documentos.

3 Técnicas computacionais aplicáveis à detecção de plágio

Este capítulo introduz diferentes técnicas de processamento de texto que podem de alguma forma auxiliar no processo de detecção de plágio.

As diferentes fases da arquitetura, que é proposta no capítulo 4 desta dissertação, necessitam do auxílio de diferentes técnicas computacionais. Algoritmos de sumarização podem ser úteis para destacar regiões do texto de maior suspeita, assim como algoritmos, assim como algoritmos de distância de edição e técnicas de pré-processamento textual podem ser úteis na detecção de similaridades entre documentos.

Em trabalhos futuros, muitas outras técnicas e algoritmos disponibilizados pela comunidade acadêmica, e outras que ainda não foram desenvolvidas, também poderão ser aplicadas a detecção de plágio utilizando a arquitetura que esta dissertação propõe. Esta dissertação apresentará apenas algumas técnicas, de modo a permitir a validação da proposta.

3.1 Sumarização de textos

Segundo Martins et al (2001) "o ponto central da sumarização é reconhecer, em um texto, o que é relevante e o que pode ser descartado, para compor um sumário". Conforme a própria definição explica, um sumário define o que há de mais importante em um texto. Destacar partes de um texto que tenham elevada relevância pode revelar sentenças ou parágrafos chaves para a indicação de outro texto como possível fonte de plágio para o documento analisado.

Diversas alternativas de algoritmos para sumarização estão disponíveis na literatura, tendo alguns desses algoritmos sido publicados há mais de cinquenta anos. Alguns dos algoritmos mais importantes já publicados podem ser encontrados no trabalho de Das & Martins (2007). Entre esses algoritmos, esta dissertação selecionou um algoritmo clássico para exemplificar, o algoritmo de sumarização de Luhn (1958).

No algoritmo de Luhn, tem-se por base a característica das linguagens escrita e falada em ideias intelectualmente associadas tendem a ser descritas por palavras fisicamente

próximas (LUHN, 1958). Dessa forma, ao calcular a importância de uma sentença, o algoritmo leva em consideração a distância entre palavras consideradas importantes.

A importância de uma palavra é dada pela sua frequência no documento ou seção de documento analisada. No entanto, palavras excessivamente frequentes e palavras pouco frequentes são consideradas *outliers* e devem ser desconsideradas, conforme demonstra a Figura 5. Na Figura 5 as palavras são ordenas de forma decrescente de frequência e as retas C e D demarcam o conjunto de palavras consideradas significativas pelo algoritmo. Palavras anteriores à reta C são desconsideradas por serem excessivamente comuns, enquanto palavras posteriores à reta D são desconsideradas por terem frequência irrelevante. Nesta abordagem palavras variantes uma das outras são consideradas iguais, como, por exemplo, as palavras: frequente, frequentemente e frequência.

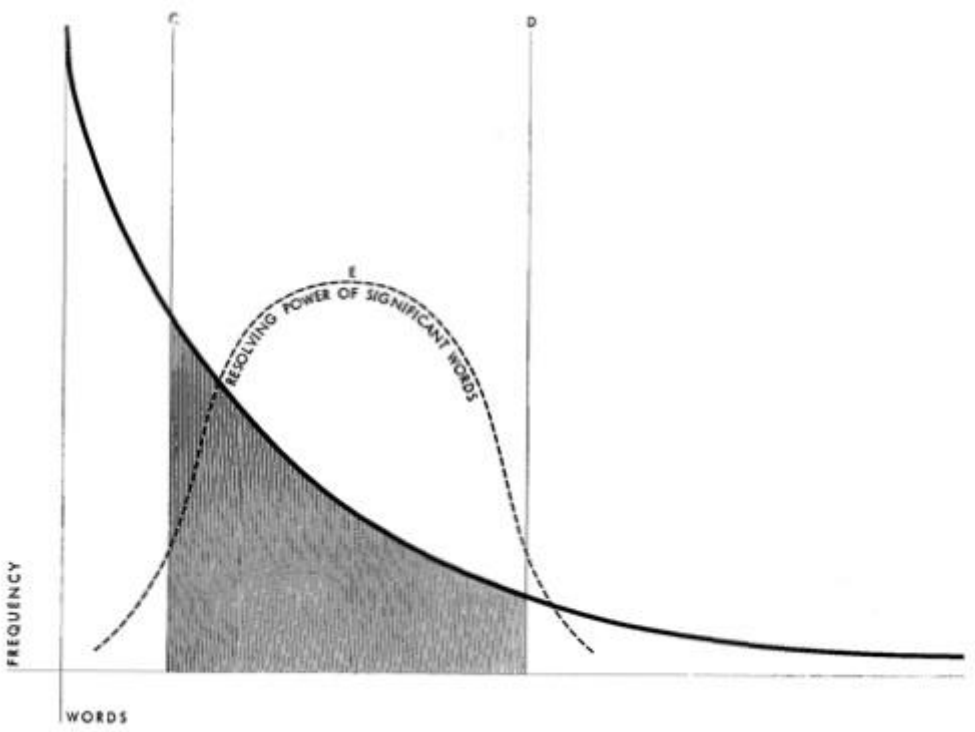


Figura 5 – Diagrama de frequência de palavras (LUHN, 1958).

Uma análise com diversos documentos mostrou que duas palavras significativas, que estejam relacionadas, não deverão ter entre elas mais do que quatro ou cinco palavras não significativas (LUHN, 1958). Dessa forma, selecionando-se duas palavras significativas que não sejam distanciadas por mais do que quatro palavras não significativas, bem como todas as palavras entre elas, calcula-se a importância da sentença considerando o número de palavras significantes nesse conjunto e a distância

entre elas. A Figura 6 apresenta uma sentença com um conjunto delimitado e utilizado no cálculo. Para o caso de mais de um conjunto de palavras serem formados, a importância da sentença é dada pelo maior valor encontrado entre os valores dos conjuntos.

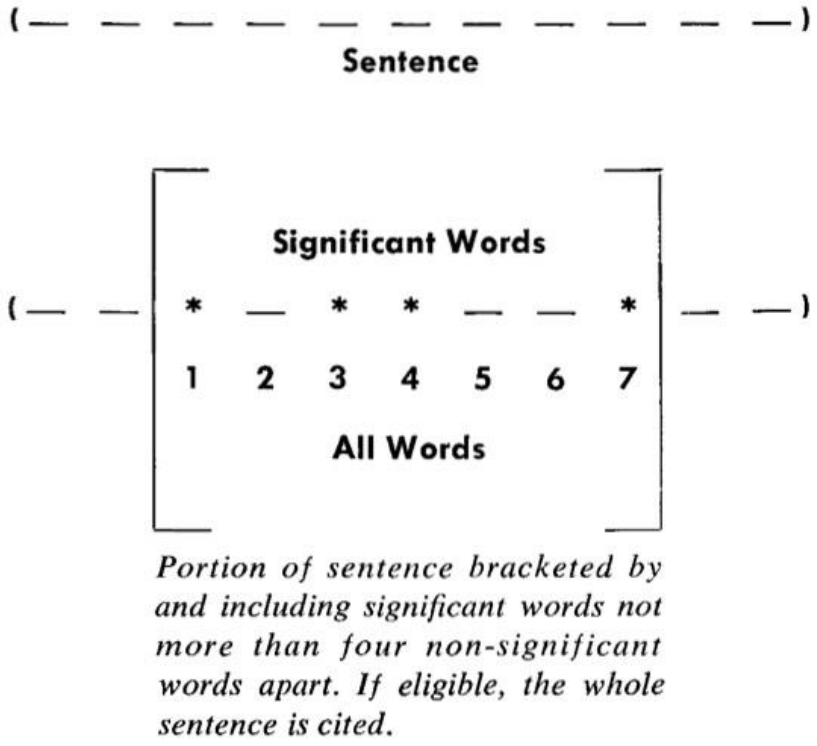


Figura 6 – Calculo da importância de uma sentença (LUHN, 1958).

A importância de uma porção delimitada de uma sentença é calculada através da fórmula abaixo:

$$importância(porção) = \frac{\sqrt{|palavrasSignificantes(porção)|}}{|palavras(porção)|}$$

, onde $|palavrasSignificantes(porção)|$ é o número de palavras significantes na porção da sentença e $|palavras(porção)|$ é o total de palavras na porção da sentença.

As sentenças mais importantes deverão ser colocadas no sumário até um limite determinado.

3.2 Distância de edição

A distância de edição ou distância de Levenshtein (LEVENSHTEIN, 1966) indica quantas inserções, deleções e substituições são necessárias para transformar uma

sequência em outra. Segundo Duarte (2011), há várias áreas nas quais essa métrica pode ser aplicada, como bioinformática, linguística, recuperação da informação, classificação de textos, erros de digitação, entre outros.

Paleo (2007) apresenta uma implementação da distância de Levenshtein utilizando programação dinâmica com complexidade de tempo e espaço O(*n*,*m*), onde *n* e *m* são os tamanhos das sequências comparadas. Esse algoritmo foi apresentado em português por Duarte (2011) na forma abaixo:

1. Preenche-se uma matriz bidimensional D, onde D[i][j] representa a distância entre o prefixo de tamanho i da primeira string S1 (de tamanho m) e o prefixo de tamanho j da segunda string S2 (de tamanho n);
2. Para i de 1 até m e j de 1 até n calcula-se D[i][j] da seguinte forma:
3. Se o caractere i de S1 é igual ao caractere j de S2 então D[i][j] = D[i – 1][j – 1];
4. Caso contrário é atribuído a D[i][j] o mínimo entre:
    a. D[i – 1][j – 1] + custo de substituição;
    b. D[i][j – 1] + custo de adição;
    c. D[i – 1][j] + custo de remoção;
5. A distância entre S1 e S2 será o valor na posição D[m][n] ao termino da execução do algoritmo;

3.3 Pré-processamento textual

Os diversos algoritmos disponibilizados para análise de textos, em geral, compartilham de uma fase de pré-processamento do texto, no qual o texto recebe um tratamento básico ou "limpeza" para que a análise tenha uma menor interferência de elementos textuais de significado pouco relevante.

Os elementos de pouca relevância irão variar de acordo com o tipo de análise executada. Por exemplo, para uma análise da estrutura do texto, os artigos podem ser relevantes, enquanto para uma análise estatística, os mesmos são tão frequentes que se tornam irrelevantes.

3.3.1 Stop words

Stop words são palavras tão comuns à escrita que para uma análise estatística de um texto se tornam irrelevantes. Como *stop words* podem ser considerados artigos,

pronomes, preposições, entre outros, dependendo do contexto em que se está trabalhando.

Hoje, já estão disponíveis na Internet diversas listas de palavras consideradas comuns, e em diversas línguas. Dentre as mais populares temos as listas de *stop words* disponibilizadas pelo Apache Lucene. O Lucene é um projeto *opensource* de uma ferramenta para auxílio à indexação, tratamento e busca em documentos textuais, que disponibiliza uma API JAVA para uso de suas funcionalidades em outras aplicações.

O Lucene já está em sua versão 3.3.0[5], tendo evoluído suas listas de *stop words* nos seus mais de dez anos de existência. O Lucene disponibiliza *stop words* em diversas línguas, estando entre elas o inglês e o português.

As *stop words* em inglês[6] do Lucene 3.3.0 são: but, be, with, such, then, for, no, will, not, are, and, their, if, this, on, into, a, or, there, in, that, they, was, is, it, an, the, as, at, these, by, to, of.

As *stop words* em português[7] do Lucene 3.3.0 são: a, ainda, alem, ambas, ambos, antes, ao, aonde, aos, apos, aquele, aqueles, as, assim, com, como, contra, contudo, cuja, cujas, cujo, cujos, da, das, de, dela, dele, deles, demais, depois, desde, desta, deste, dispoe, dispoem, diversa, diversas, diversos, do, dos, durante, e, ela, elas, ele, eles, em, entao, entre, essa, essas, esse, esses, esta, estas, este, estes, ha, isso, isto, logo, mais, mas, mediante, menos, mesma, mesmas, mesmo, mesmos, na, nas, nao, nas, nem, nesse, neste, nos, o, os, ou, outra, outras, outro, outros, pelas, pelas, pelo, pelos, perante, pois, por, porque, portanto, proprio, propios, quais, qual, qualquer, quando, quanto, que, quem, quer, se, seja, sem, sendo, seu, seus, sob, sobre, sua, suas, tal, tambem, teu, teus, toda, todas, todo, todos, tua, tuas, tudo, um, uma, umas, uns.

3.3.2 Stemming

Radicalização ou *stemming* consiste em, dada uma palavra, extrair o seu radical. O intuito da radicalização é tornar equivalentes palavras derivadas de uma mesma palavra, e que possivelmente terão o mesmo sentido. Por exemplo, as palavras “animar”, “animação” e “animem” possuem o radical “anim”.

[5] Consulta em 19 de agosto de 2011 ao website do projeto (http://lucene.apache.org).
[6] Obtida através da classe JAVA org.apache.lucene.analysis.StopAnalyzer (Lucene 3.3.0).
[7] Obtida através da classe JAVA org.apache.lucene.analysis.br.BrazilianAnalyzer (Lucene 3.3.0).

Através do *stemming* é possível tornar equivalentes palavras que compartilhem um mesmo radical, e dessa forma atribuir maior peso a esse grupo de termos dentro de um texto. Esses dados poderão ser utilizados em análises estatísticas que detectem similaridades entre textos.

3.4 Bag of words

Bag of words é uma abordagem estatística para comparação de documentos textuais, na qual cada documento é representado por um conjunto de termos e suas frequências. Nessa abordagem a ordem dos termos no documento não é considerada, tendo dessa forma a semântica do documento desprezada.

Durante a construção do *bag of words* de um documento, normalmente, o documento é pré-processado, sendo eliminadas as *stop words* e as demais palavras são transformadas em seus radicais através de *stemming*.

3.5 MySQL Fulltext

O sistema gerenciador de banco de dados MySQL[8] disponibiliza uma forma de busca textual alternativa ao tradicional comando *like* do SQL. Essa busca implementa a técnica *full-text* para indexação e busca textual (ORACLE CORPORATION, 2011a).

O MySQL Fulltext permite a criação de índices que agreguem um ou mais campos textuais de uma tabela. Ao executar uma consulta utilizando *full-text* na cláusula *where*, o MySQL retorna os registros por ordem de relevância**.** Segundo Varella (2007), essa relevância é dada pelo somatório das relevâncias de cada termo da consulta no registro em questão, e a relevância de um termo é dada pelo seu peso no registro multiplicado pela sua freqüência na *string* de consulta, conforme mostra a fórmula abaixo:

$$R(q,d) = \sum_{t \in d} weight(t,d) * freq(t,q)$$

, onde *weight(t,d)* indica o peso de um termo *t* em um documento (registro) *d*, *e freq(t,q)* indica a freqüência de um termo *t* em uma consulta *q*.

Segundo Oracle Corporation (2011b), o peso de um termo em um documento será dado pela fórmula abaixo:

[8] Disponível em http://dev.mysql.com/

$$w = \frac{log(dtf) + 1}{sumdtf} * \frac{U}{1 + 0.0115 * U} * \frac{log((N - nf)}{nf}$$

, onde "*dtf* é o número de vezes que um termo aparece no documento, *sumdtf* é a soma de (log(dtf)+1) de todos os termos do documento, *U* é número de termos únicos no documento, *N* é o total de documentos na base e *nf* é o total de documentos que contêm o termo" (ORACLE CORPORATION, 2011b) (tradução livre).

O uso do MySQL Fulltext na recuperação de dados textuais pode facilitar a seleção dos dados a serem processados por um software. Uma vez que o processamento textual costuma ter grande custo computacional, e, muitas vezes, inviabiliza o processamento de toda uma base de dados, o *full-text* pode ser utilizado para restringir o processamento aos documentos mais relevantes.

3.6 WordNet

WordNet é uma base de dados léxica desenvolvida para ser utilizada por programas de computador (MILLER, 1995). Ela possui um extenso vocabulário em inglês, seus significados e os relacionamentos entre palavras.

Miller (1995) define palavra como um par (*forma, sentido*), onde forma é uma sequência de caracteres que, pertencentes a um alfabeto finito, que representa uma palavra e sentido é um significado atribuído a mesma.

A WordNet, hoje, possui mais de 155 mil palavras únicas, entre nomes, verbos, adjetivos e advérbios, e mais de 117 mil *synsets* (THE TRUSTEES OF PRINCETON UNIVERSITY, 2011). Um *synset* ou conjunto de sinônimos é uma representação de um conceito léxico, utilizado para agrupar palavras que compartilhem de um mesmo sentido (MILLER, 1995).

Ainda segundo Miller (1995), a WordNet define os seguintes relacionamentos semânticos:

- *Synonymy* é um relacionamento entre palavras que pertençam a um mesmo *synset*, ou seja, que compartilham de um mesmo sentido;
- *Antonymy*: é um relacionamento entre palavras com significados opostos;

- *Hyponymy* (sub-nome) e *hypernymy* (super-nome) relacionam *synsets* de forma hierárquica. Como exemplo, temos que *canine* é um *hypernymy* de *dog*, onde um cão é um tipo de canino;

- *Meronymy* é um relacionamento no qual o que é representado por uma palavra compõe o que outra palavra representa, enquanto *holonymy* é uma relação na qual a representa de uma palavra é composta pela representação de outra. Exemplo: *accelarator* é um *meronymy* de *car*, enquanto *car* é um *holonymy* de *accelarator*;

- *Troponymy* é utilizado para relacionar verbos. Leung & Chan (2007) definem que um verbo Y é *troponymy* de um verbo X, se a atividade de Y é uma maneira de executar a atividade de X. Exemplo: *lisp* é um *troponymy* de *talk*;

- *Entailment* é uma relação na qual a execução da atividade de um verbo implica na execução da atividade de outro verbo. Exemplo: *walk* é um *entailment* de *step*.

A Figura 7 apresenta as relações semânticas, as categorias de palavras as quais se aplicam e alguns exemplos das mesmas.

Semantic Relation	Syntactic Category	Examples
Synonymy (similar)	N, V, Aj, Av	pipe, tube rise, ascend sad, unhappy rapidly, speedily
Antonymy (opposite)	Aj, Av, (N, V)	wet, dry powerful, powerless friendly, unfriendly rapidly, slowly
Hyponymy (subordinate)	N	sugar maple, maple maple, tree tree, plant
Meronymy (part)	N	brim, hat gin, martini ship, fleet
Troponomy (manner)	V	march, walk whisper, speak
Entailment	V	drive, ride divorce, marry

Note: *N = Nouns* *Aj = Adjectives* *V = Verbs* *Av = Adverbs*

Figura 7 – Relações semânticas na WordNet (MILLER, 1995)

Uma base de dados léxica é útil para atribuir significado aos termos de um documento, bem como identificar os relacionamentos entre os mesmos. Essa utilidade abre muitas possibilidades de aplicação na comparação de documentos e detecção de similaridades entre os mesmos.

4 Proposta de arquitetura para um sistema de detecção de plágio multi-algoritmo

Neste capítulo é apresentada a proposta principal deste trabalho, uma arquitetura para um sistema de detecção de plágio multi-algoritmo. Essa arquitetura se propõe a modelar o problema da detecção de plágio, dividindo-o em problemas menores e disponibilizando interfaces para que diversas propostas de solução, para cada um desses problemas, possam ser implementadas e testadas, tornando a resolução desses problemas, e do problema maior detecção de plágio, mais colaborativa, para que outros trabalhos possam continuar este, e a aplicação de algoritmos de diversas áreas seja facilitada.

O capítulo inicia propondo o problema e uma solução para enfrentá-lo. Mais adiante, é apresentado um modelo teórico para a arquitetura da ferramenta, bem como para a estrutura dos documentos a serem analisados pela mesma.

4.1 Solução proposta: Dividir para conquistar

Esta dissertação propõe uma ferramenta de auxílio à detecção de plágio na qual é possível submeter documentos e solicitar uma ou mais análises de cada documento, customizáveis através da escolha dos algoritmos a serem utilizados.

A ferramenta propicia um ambiente para testes de diversos algoritmos no processo de detecção de plágio. Este trabalho, além de apresentar a ferramenta, sua arquitetura e a implementação de um protótipo da mesma, também apresenta os testes feitos com diversos algoritmos e os resultados obtidos com cada um deles, bem como as melhorias obtidas conforme os algoritmos foram evoluídos.

Nesta proposta, o problema da detecção de plágio foi dividido em cinco problemas de menor proporção: definição de consultas, recuperação de documentos, detecção de similaridades, refinamento de comparações e formulação de índice de plágio. Para cada um desses problemas existem inúmeras soluções, e cada solução gera um resultado que pode ser aceitável ou não.

A arquitetura proposta deve promover uma plataforma para construção de soluções para cada um desses problemas. A combinação de soluções para os cinco problemas deverá levar a uma solução para o problema da detecção de plágio, que por sua vez deverá ter seus resultados avaliados.

4.1.1 Definição de consultas

Diversos são os repositórios nos quais podem estar presentes documentos fonte para plágio. Esses repositórios podem variar desde pastas no computador de um usuário, passando por avançados Sistemas Gerenciadores de Banco de Dados e ferramentas avançadas de buscas de documentos como o Lucene[9], até a própria World Wide Web.

Cada um desses repositórios possui uma interface ou um meio de recuperação dos documentos nos quais se está interessado. Essas interfaces proveem meios para consulta ao repositório, de modo a identificar e recuperar documentos que possam interessar aos seus usuários.

A necessidade em usar consultas está, principalmente, ligada ao fato de nem sempre ser viável em uma análise de plágio comparar o documento suspeito com todos os documentos do(s) repositório(s).

No entanto, nem sempre é simples formular uma consulta. No caso de uma análise para detecção de plágio, quase sempre, a única informação disponível é o próprio documento analisado. Um exemplo de consulta para uma ferramenta como o Lucene poderia ser o texto completo do próprio documento, juntamente com alguns parâmetros de configuração que o Lucene eventualmente necessite.

As consultas para detecção de plágio podem ser mais complexas do que consultas habituais, uma vez que um documento pode ser uma montagem com textos de diversos documentos e, ainda, ter cada uma de suas sentenças modificadas, através dos métodos apresentados na seção 2.1. O que implica na necessidade de soluções mais sofisticadas para a geração dessas consultas.

4.1.2 Recuperação de documentos

Inúmeras podem ser as fontes utilizadas por plagiadores. Podem ser utilizados desde trabalhos de colegas de uma classe de estudos, que poderão estar disponíveis em uma

[9] Apache Lucene – Disponível em http://lucence.apache.org

pasta no computador do professor, passando por bases de artigos e chegando mesmo a documentos disponibilizados na Web e encontrados através de ferramentas de busca.

Uma ferramenta para detecção de plágio deve ser capaz de consultar todas as fontes de documentos suspeitas. A proposta dessa dissertação provê uma interface para a implementação de métodos de acesso as diversas fontes disponíveis.

4.1.3 Detecção de similaridades

O terceiro problema a ser resolvido, ocorre após se obter os documentos a serem comparados. Para cada documento recuperado, deve-se compará-lo com o documento analisado e detectar similaridades que possam vir a indicar plágio.

A detecção de similaridades pode ser feita com o uso de inúmeros algoritmos. Desde algoritmos estatísticos até algoritmos de comparação semântica. Cada algoritmo tem eficácia e uma complexidade computacional distintas, e consequentemente tempo de retorno diferentes.

As similaridades são analisadas a partir da estrutura de documento definida na seção 4.2.2. Uma similaridade é definida por um valor real, indo de zero a um, e de um trecho (sentença, parágrafo, seção ou documento completo) para cada um dos documentos comparados. Uma comparação entre dois documentos pode gerar inúmeras similaridades, dependendo do algoritmo utilizado. Um algoritmo que compare cada um dos *n* parágrafos de um documento com os *m* parágrafos do outro poderá gerar, por exemplo, até *n* vezes *m* similaridades.

4.1.4 Refinamento de comparações

Os algoritmos de comparação utilizam como base a estrutura de documento definida na seção 4.2.2, no entanto, dividir os documentos conforme a estrutura definida não é uma tarefa simples. Os documentos podem estar representados em diversos formatos, e nem sempre esses formatos demarcam bem o início e fim de um parágrafo, por exemplo. Essa situação acarreta em erros de divisão do documento na estrutura descrita e, por consequência, pode atrapalhar a detecção de similaridades.

Para amenizar o erro descrito acima, é utilizado um refinamento, no qual, a partir de similaridades encontradas entre os documentos, se utiliza algoritmos para reestruturar os documentos e executar novas comparações. O intuito desses algoritmos é resolver o quarto problema proposto: reestruturação de documentos.

4.1.5 Índice de plágio

O último problema a resolver é como determinar se um documento é plágio de outro ou não. Este trabalho, após verificar diversas ferramentas, como descrito na seção 2.4, conclui que, hoje, a solução mais viável para determinar se um documento é plágio de outro, ou não, é através da análise humana, e que os softwares seriam úteis para demonstrar suspeitas sobre documentos ou partes deles. Quanto maior o índice de acertos e a precisão em indicar suspeita de um plágio, maior a facilidade trazida ao trabalho humano e, consequentemente, melhor será avaliada a ferramenta.

É possível que um dia o índice de acertos de uma ferramenta de suporte a detecção de plágio chegue a um valor próximo do índice de acertos dos seres humanos ou até mesmo ao valor de cem por cento. Nesse dia, talvez se possa abrir mão da análise humana. Enquanto esse dia não chega, deve-se aperfeiçoar cada vez mais, tanto os algoritmos de detecção de similaridades, quanto os cálculos para determinar os índices de plágio.

Índice de plágio é um número real de zero a um, cujo valor determina a probabilidade de um documento, ou parte de um documento, ser plágio de outro documento, ou de parte dele. O índice de plágio pode ser desde uma convenção de que o índice será a maior similaridade encontrada entre um documento e outro, até um cálculo que inclua todas as similaridades encontradas. Em geral, o índice de plágio irá indicar o quão suspeito um documento, ou parte dele, é.

4.2 Modelo Teórico

4.2.1 Arquitetura

A ferramenta de auxílio à detecção de plágio proposta divide o trabalho de detecção de plágio em cinco fases, conforme pode ser viso na Figura 8. As cinco fases são: extração de consultas, recuperação de documentos, análise de documentos, análise de plágio e geração de resultados.

As quatro primeiras fases são responsáveis por resolver os cinco problemas apresentados na seção 4.1, enquanto a última fase finaliza a análise com a apresentação de relatórios.

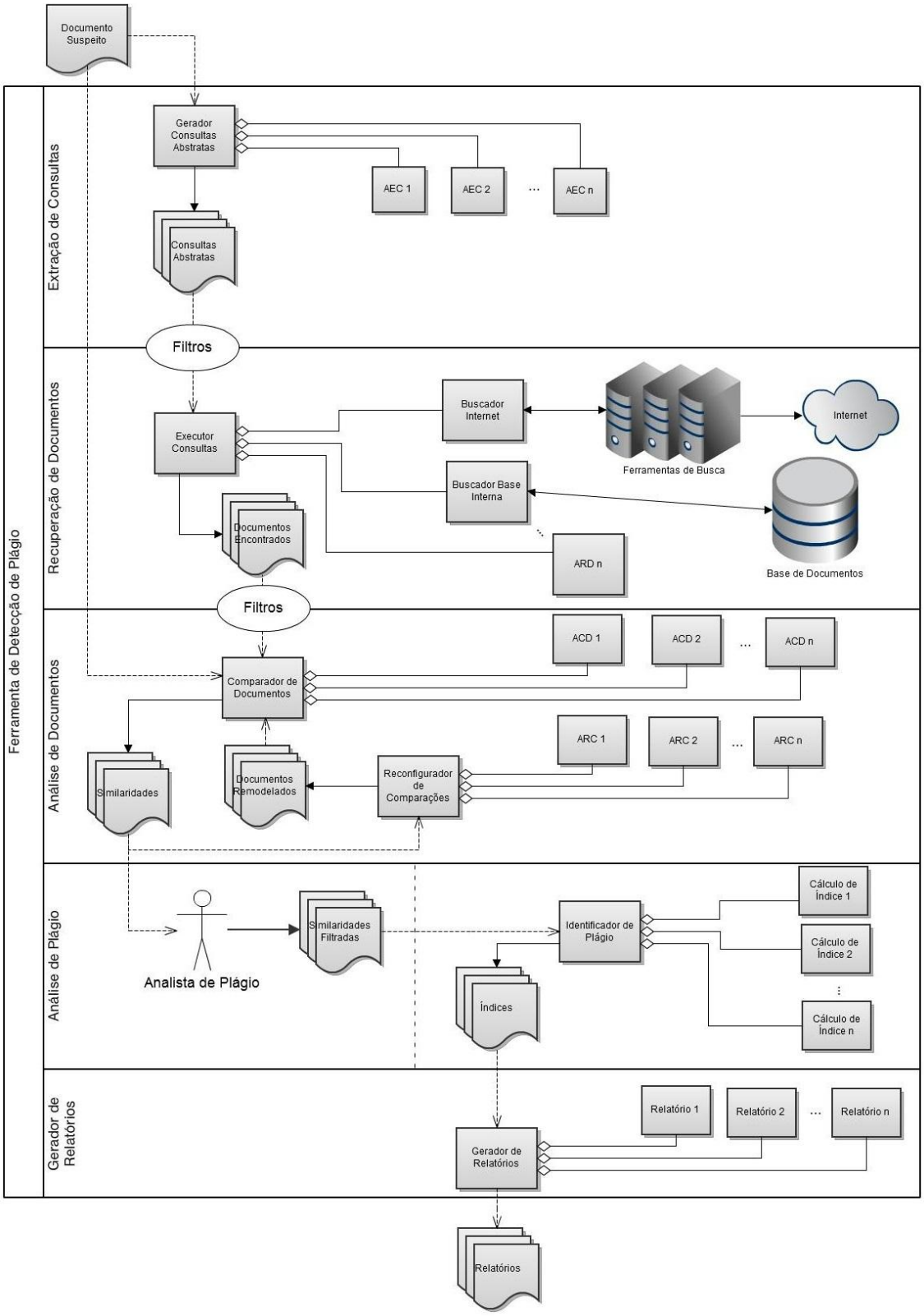


Figura 8 – Arquitetura da ferramenta de detecção de plágio.

Na fase de extração de consultas (Figura 9), o Gerador de Consultas Abstratas utiliza os Algoritmos de Extração de Consultas (AEC) para gerar as chamadas Consultas Abstratas *ou Abstract Queries*, que são textos gerados a partir da manipulação de parte

ou mesmo do texto completo do documento suspeito, podendo sofrer alterações ou não, para serem utilizados pelos algoritmos de recuperação de documentos. As Consultas Abstratas podem ser desde textos simples, que podem ser buscados pelo Google, por exemplo, até complexos textos em XML[10] que poderão ser processados por complexos algoritmos de recuperação de documentos.

Ao fim da fase, as Consultas Abstratas são submetidas a um filtro, que irá eliminar as consultas que seguirem um ou mais padrões pré-cadastrados como consultas de pouco interesse, por obterem resultados genéricos e de pouca aderência ao assunto específico do texto. Como exemplo de consultas filtradas, pode-se citar consultas que só tenham caracteres numéricos ou que contenham apenas palavras ou siglas conhecidas, como a sigla "UFRJ". Os padrões utilizados nos filtros deverão ser selecionados ao se submeter uma análise.

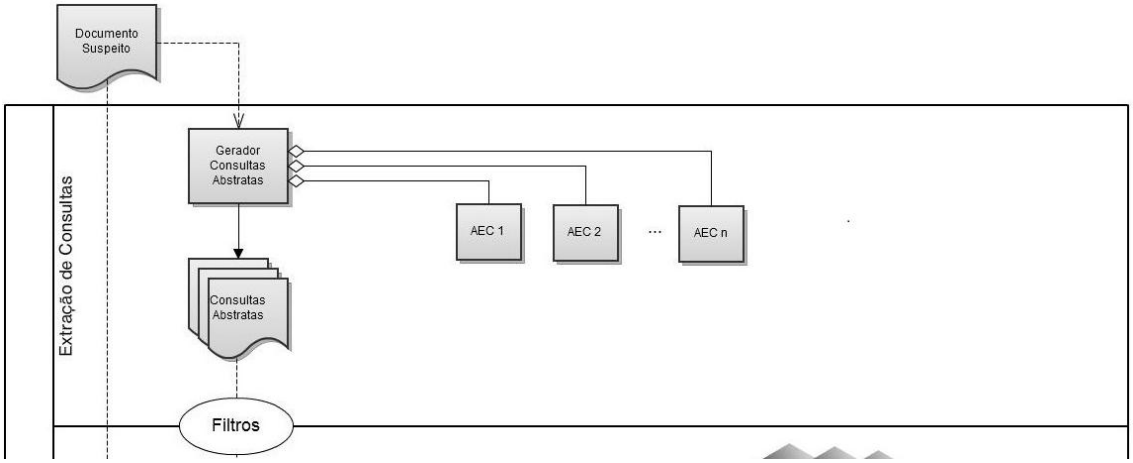


Figura 9 - Extração de Consultas

Na fase de recuperação de documentos (Figura 10), o módulo Executor de Consultas utiliza os Algoritmos de Recuperação de Documentos (ARD) para, através de uma Consulta Abstrata, recuperar um conjunto de documentos a ser comparado com o documento em análise (documento suspeito). Os ARD's podem estar relacionados a diferentes bases de documentos, desde repositórios de documentos e bancos de dados locais até documentos disponíveis na Web.

Os documentos encontrados são então filtrados, para depois serem submetidos à fase seguinte da análise. Os filtros utilizados, em geral, são documentos que reconhecidamente são o documento submetido à análise. Esses documentos podem ter

[10] *Extensible Markup Language* ou Linguagem de Marcação Extensível é utilizada para atribuir significado a um texto.

sido recuperados devido ao fato do documento analisado estar publicado em um ou mais lugares acessíveis pelos ARD's. As informações utilizadas nos filtros devem ser informadas pelo usuário que submeteu a análise.

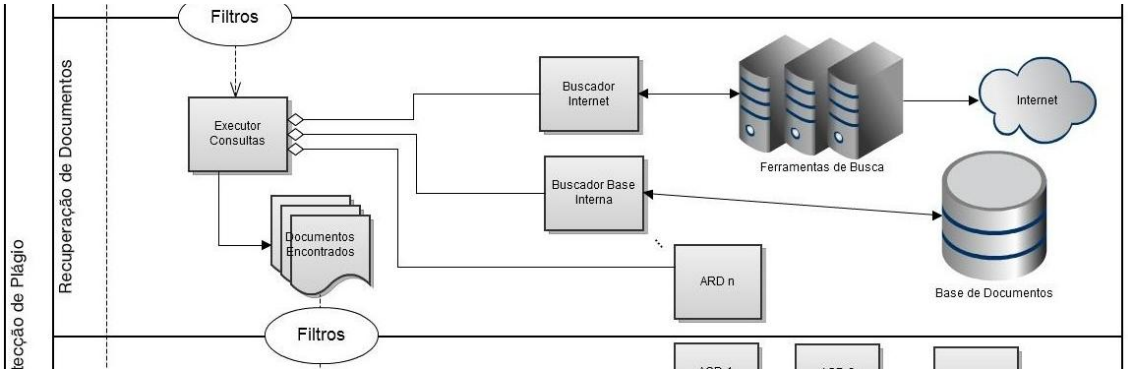


Figura 10 - Recuperação de documentos

Na fase de análise de documentos (Figura 11), os Algoritmos de Comparação de Documentos (ACD) comparam o documento suspeito com um documento recuperado por um ARD, através do módulo Comparador de Documentos. Esses algoritmos retornam similaridades encontradas entre os documentos comparados. Cada similaridade é identificada por um conjunto de três parâmetros: parte do documento suspeito, parte do documento encontrado e a similaridade entre essas partes, sendo essa última um número real entre zero e um.

O módulo Reconfigurador de Comparações é responsável por, a partir das similaridades encontradas, sugerir novas redistribuições dos textos de um documento conforme o modelo de documentos apresentado na seção 4.2.2. Esse módulo se torna necessário devido a dificuldade de se determinar o início e fim de um parágrafo, sentença ou seção de um documento, que pode ser erradamente dividido pela ferramenta devido a erros de interpretação de arquivos do formato PDF[11], entre outros.

[11] Portable Document Format (PDF) é um formato de arquivo muito utilizado para disponibilização de documentos.

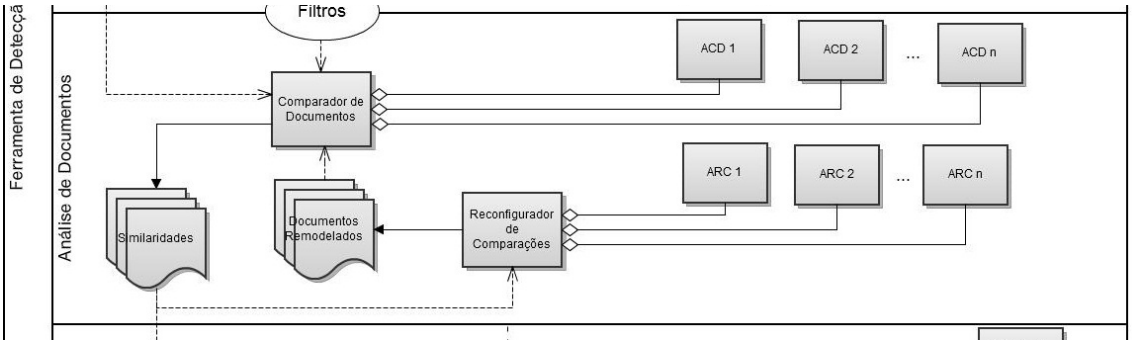


Figura 11 - Análise de documentos

Durante a fase Análise de Plágio (Figura 12), as similaridades encontradas são apresentadas a um Analista de Plágio, um ser humano com conhecimentos sobre o conceito de plágio e suas diferentes formas. O Analista é responsável por filtrar entre as similaridades, aquelas que, em sua opinião, são plágio.

As similaridades filtradas são então utilizadas para o cálculo dos índices de plágio do documento. O componente Identificador de Plágio pode executar um ou mais algoritmos de Cálculo de Índice. Os índices encontrados são então liberados para uso nos relatórios.

O trabalho de filtrar as similaridades, feito pelo Analista de Plágio, também pode ser visto como uma avaliação da similaridade, visto que a ferramenta deverá permitir uma análise subjetiva, para o caso de incerteza do analista, e que poderá influenciar o cálculo de um ou mais indicadores.

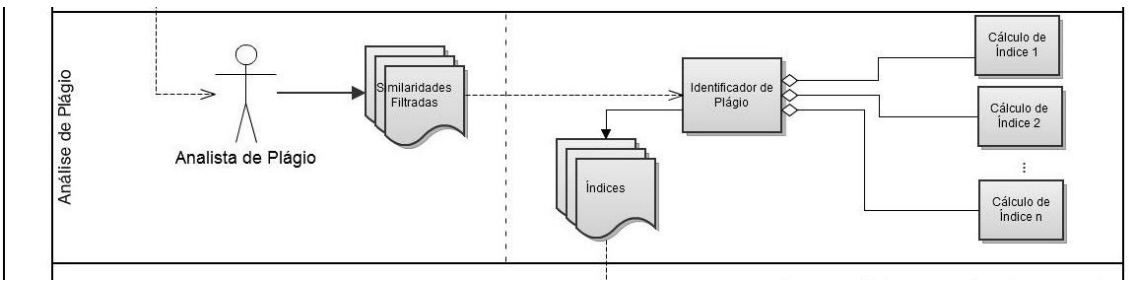


Figura 12 - Análise de plágio

O módulo Gerador de Relatórios (Figura 13) é responsável por, através dos índices de plágio e das similaridades associadas a eles, criar relatórios que poderão ser apresentados como prova de um plágio.

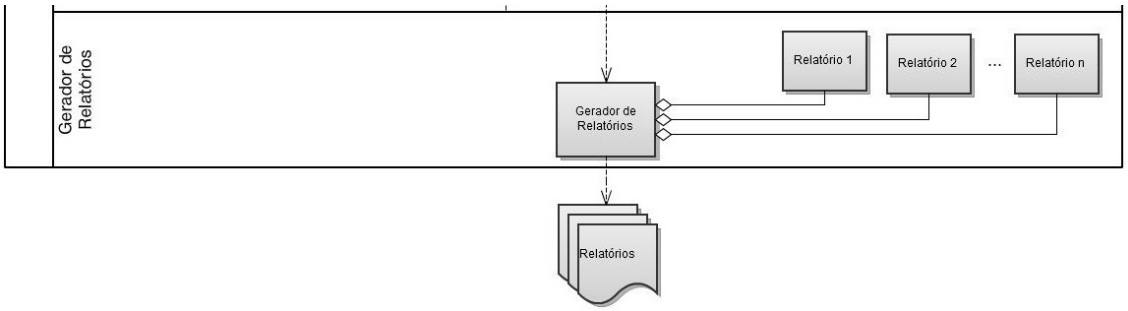


Figura 13 - Gerador de relatórios

4.2.2 Modelos de documentos

Para os propósitos deste trabalho foi especificado um modelo próprio para documentos que abrangesse a granularidade textual necessária a análise de plágio (granularidade de sentença), bem como representasse a relação entre as diferentes partes de um documento, incluindo a sua bibliografia e a relação de *referência*.

A Figura 14 apresenta o modelo para documentos definido. Nele é definido que um documento possui um ou mais autores e é composto por uma ou mais seções. Cada seção, por sua vez, pode ter subseções e parágrafos, onde parágrafos podem ser vistos como conjuntos de sentenças.

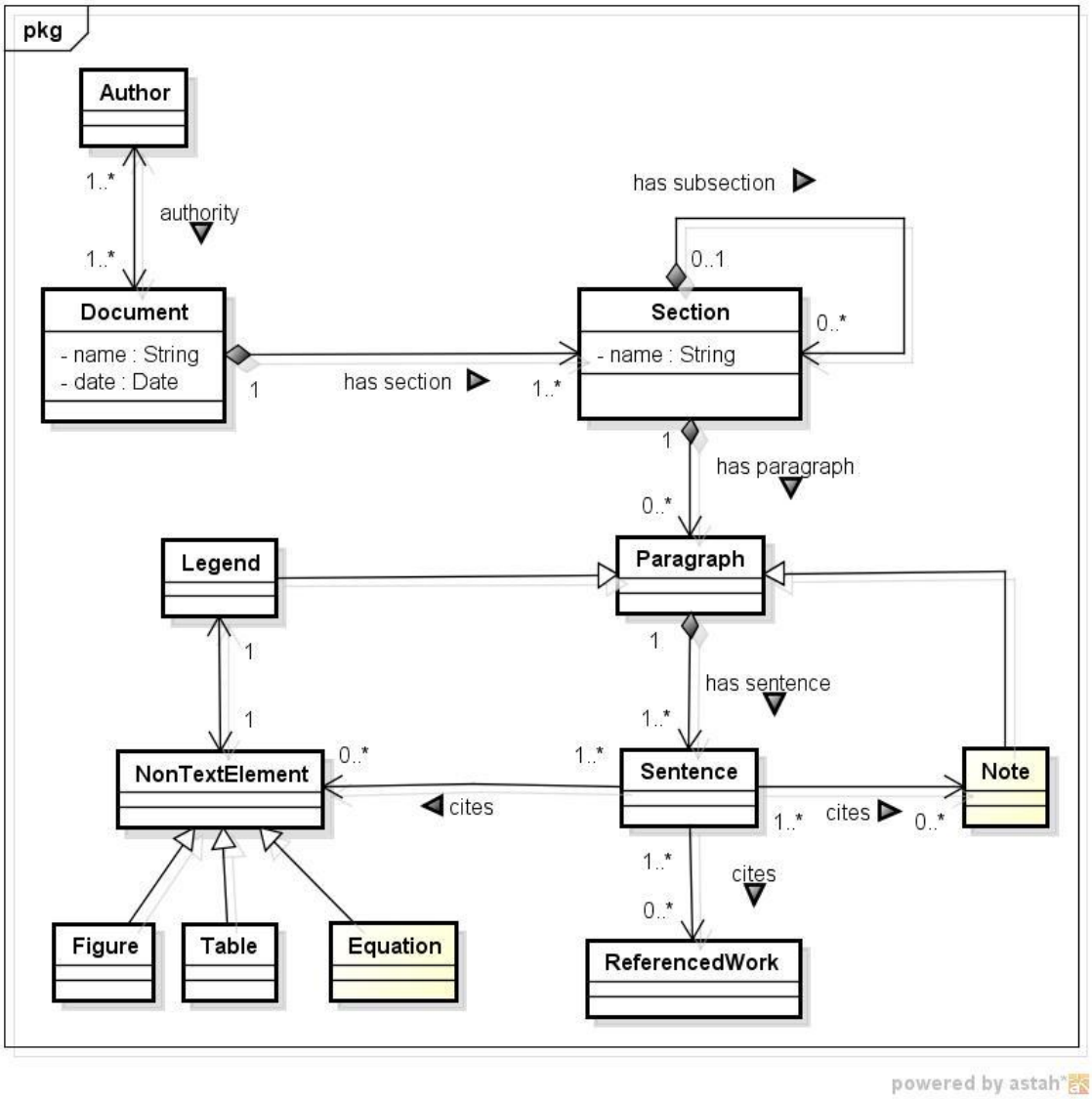


Figura 14 - Modelo geral de documentos.

Uma sentença é definida neste trabalho como um conjunto de palavras organizadas conforme as regras gramaticais da língua em que a sentença se apresenta, podendo ser a língua do documento ou não. Uma sentença pode conter referências para elementos não textuais, como figuras, tabelas e equações, para outros trabalhos, geralmente listados em uma seção de referências bibliográficas, e para notas de rodapé.

Um documento também é composto por elementos não textuais, trabalhos referenciados e notas de rodapé. O primeiro pode ser especializado em entidades como figuras, tabelas e equações. Cada elemento não textual possui uma legenda, uma especialização de um Parágrafo, que tem como função identificá-lo e, caso necessário, explicá-lo e/ou acrescentar-lhe mais informações.

Trabalhos referenciados identificam outros documentos que tenham sido utilizados no desenvolvimento do documento em questão.

Notas de Rodapé são informações extras acrescidas ao documento, localizadas ao final de uma página, de modo a facilitar o entendimento de certo trecho do texto na mesma página.

4.2.3 Teste de mesa da proposta

Para o texto de mesa foi produzido um documento, através do uso de plágio, a partir de dois textos sobre Lógica difusa, ou lógica *fuzzy*, disponíveis na Wikipédia[12].

Na produção do plágio foram utilizadas algumas das técnicas apresentadas na 2.1, como *copy&paste*, tradução e mudança gramatical. O documento produzido é apresentado abaixo.

[12] Textos disponíveis em http://pt.wikipedia.org/wiki/Lógica_difusa e http://en.wikipedia.org/wiki/Fuzzy_logic.

Rio de Janeiro, 27 de setembro de 2009

Universidade Federal do Rio de Janeiro

Departamento de Ciência da Computação

Aluno: José do Plágio

Trabalho sobre Lógica Fuzzy

A lógica difusa ou lógica fuzzy é uma extensão da lógica booleana que admite valores lógicos intermediários entre o FALSO (0) e o VERDADEIRO (1); por exemplo o valor médio 'TALVEZ' (0,5). Isto significa que um valor lógico difuso é um valor qualquer no intervalo de valores entre 0 e 1. Este tipo de lógica engloba de certa forma conceitos estatísticos principalmente na área de Inferência.

A lógica fuzzy começou com a proposta de 1965, a teoria dos conjuntos fuzzy por Lotfi Zadeh. Embora a lógica fuzzy tenha sido aplicada a muitos campos, a partir da teoria de controle de inteligência artificial, ainda permanece controverso entre a maioria dos especialistas em estatística, que preferem a lógica bayesiana, e alguns engenheiros de controle, que preferem a tradicional lógica bivalente.

As implementações da lógica difusa permitem que estados indeterminados possam ser tratados por ferramentas de controle. Dessa forma, é possível avaliar conceitos não-quantificáveis, como, por exemplo, avaliar a temperatura (quente, morno, médio, etc...) e o sentimento de felicidade (radiante, feliz, apático, triste...)

A lógica difusa deve ser vista mais como uma área de pesquisa sobre tratamento de incerteza, ou como um conjunto de modelos matemáticos dedicados ao tratamento da incerteza, do que como uma lógica. A lógica difusa é normalmente associada ao uso de uma teoria proposta por Lukasiewicz, a teroria de conjuntos difusos.

A lógica booleana é normalmente referenciada como lógica nítida, ao se trabalhar com lógica difusa.

Muitos pesquisadores de versões booleanas de lógica não aceitam a lógica difusa como uma verdadeira lógica, no sentido em que aceitam, por exemplo, a lógica modal.

Essa não aceitação pode ser associada a diferentes fatos, entre eles o fato de muitos modelos permitirem soluções aproximadas que não correspondem a uma "verdade" lógica.

O primeiro parágrafo foi gerado a partir de *copy & paste*. O segundo parágrafo foi gerado a partir de tradução, para tal utilizou-se a ferramenta Google Trandutor[13], seguido de alguns ajustes humanos e mudanças gramaticais. No terceiro e no quarto parágrafos foram utilizadas mudanças por sinônimos, alterações gramaticais e deleção de termos. O quinto parágrafo é um parafraseamento, enquanto o sexto e o sétimo

[13] Disponível em http://translate.google.com

parágrafos são a divisão de um único parágrafo usado como fonte, com uma pequena mudança no vocabulário.

4.2.3.1 Geração de consultas

É possível utilizar mais de uma técnica, se assim for interessante. Nesse caso, usaremos duas técnicas simples, a extração de parágrafos e a tradução de parágrafos do português para o inglês com o uso do Google Tradutor.

As técnicas devem gerar consultas que atendam aos formatos de consultas esperados pelos algoritmos de execução de consulta. Neste teste de mesa utilizaremos como consulta abstrata um texto simples. No entanto, poderíamos passar complexos arquivos XML que passassem outras informações aos algoritmos, como o tema (lógica fuzzy, computação, etc.) a qual o texto pertence, por exemplo.

As consultas geradas podem ser visualizadas na Tabela 3.

Tabela 3 – Consultas geradas.

ID	**Texto Consulta**	**ID**	**Texto Consulta**
1	*Rio de Janeiro, 27 de setembro de 2009*	13	*Rio de Janeiro, September 27, 2009*
2	*Universidade Federal do Rio de Janeiro*	14	*Federal University of Rio de Janeiro*
3	*Departamento de Ciência da Computação*	15	*Department of Computer Science*
4	*Aluno: José do Plágio*	16	*Student: Joseph of Plagiarism*
5	*Trabalho sobre Lógica Fuzzy*	17	*Work on Fuzzy Logic*
6	*A lógica difusa ou lógica fuzzy é uma extensão da lógica booleana que admite valores lógicos intermediários entre o FALSO (0) e o VERDADEIRO (1); por exemplo o valor médio 'TALVEZ' (0,5). Isto significa que um valor lógico difuso é um valor qualquer no intervalo de valores entre 0 e 1. Este tipo de lógica engloba de certa forma conceitos estatísticos principalmente na área de Inferência.*	18	*Fuzzy logic or fuzzy logic is an extension of Boolean logic that allows intermediate values between the logical FALSE (0) and TRUE (1), eg the average 'MAYBE' (0.5). This means that a fuzzy logic value is any value in the range of values between 0 and 1. This type of logic includes statistical concepts in a way particularly in the area of inference.*
7	*A lógica fuzzy começou com a proposta de 1965, a teoria dos conjuntos fuzzy por Lotfi Zadeh.*	19	*The fuzzy logic began with the 1965 draft, the theory of fuzzy sets by Lotfi Zadeh. Although fuzzy logic*

	Embora a lógica fuzzy tenha sido aplicada a muitos campos, a partir da teoria de controle de inteligência artificial, ainda permanece controverso entre a maioria dos especialistas em estatística, que preferem a lógica bayesiana, e alguns engenheiros de controle, que preferem a tradicional lógica bivalente.		*has been applied to many fields, from the control theory of artificial intelligence, remains controversial among most statisticians, who prefer the Bayesian logic, and some control engineers, who prefer the traditional bivalent logic.*
8	*As implementações da lógica difusa permitem que estados indeterminados possam ser tratados por ferramentas de controle. Dessa forma, é possível avaliar conceitos não-quantificáveis, como, por exemplo, avaliar a temperatura (quente, morno, médio, etc...) e o sentimento de felicidade (radiante, feliz, apático, triste...)*	20	*The implementations of fuzzy logic allow indeterminate states can be treated by control tools. Thus, it is possible to evaluate non-quantifiable concepts, such as, for example, to evaluate the temperature (hot, warm, medium, etc ...) and the feeling of happiness (bright, happy, apathetic, sad ...)*
9	*A lógica difusa deve ser vista mais como uma área de pesquisa sobre tratamento de incerteza, ou como um conjunto de modelos matemáticos dedicados ao tratamento da incerteza, do que como uma lógica. A lógica difusa é normalmente associada ao uso de uma teoria proposta por Lukasiewicz, a teroria de conjuntos difusos.*	21	*Fuzzy logic should be seen more as an area of research on treatment of uncertainty, or as a set of mathematical models dedicated to the treatment of uncertainty, rather than as a logic. Fuzzy logic is usually associated with the use of a theory proposed by Lukasiewicz, the teroris of fuzzy sets.*
10	*A lógica booleana é normalmente referenciada como lógica nítida, ao se trabalhar com lógica difusa.*	22	*Boolean logic is generally referred to as a clear logic, when working with fuzzy logic.*
11	*Muitos pesquisadores de versões booleanas de lógica não aceitam a lógica difusa como uma verdadeira lógica, no sentido em que aceitam, por exemplo, a lógica modal.*	23	*Many researchers versions of Boolean logic does not accept the logic of fuzzy logic as true, in that support, for example, modal logic.*
12	*Essa não aceitação pode ser associada a diferentes fatos, entre eles o fato de muitos modelos permitirem soluções aproximadas que não correspondem a uma "verdade" lógica.*	24	*This rejection may be associated with different events, including the fact that many models allow approximate solutions which do not correspond to a "true"logic.*

Na Tabela 3, temos a consulta, de identificador igual a 2, com o texto “Universidade Federal do Rio de Janeiro” (UFRJ) e sua tradução para o inglês, através do Google

Tradutor. Uma consulta a partir desse texto pode ser considerada genérica, visto que, a UFRJ é instituição de ensino, a qual uma variada gama de assuntos está relacionada, e documentos em diversos temas podem ser encontrados.

Para evitar o uso de recursos computacionais com consultas e comparações desnecessárias, é aconselhada a definição de filtros para uso na análise. Esses filtros podem englobar desde filtros que eliminem consultas que, em seu texto, só contenham *stop words* e termos selecionadas pelo usuário como genéricos, até complexas expressões regulares, que por sua vez também podem referenciar um conjunto de termos.

Para a análise desse teste de mesa, foram definidos os filtros apresentados na Tabela 4. Os filtros, bem como outras definições necessárias, foram representados e formato XML. O elemento "filtro" define uma expressão regular que identifica um padrão para textos de consultas abstratas que deverão ser desconsideradas. De forma complementar, o elemento "dicionário" define um conjunto de termos que poderão ser utilizados nesses filtros. No exemplo da Tabela 4, foram definidos dicionários com nomes de cidades, termos comuns a trabalhos de um departamento de uma universidade e importados dicionários que contém *stopwords* e os nomes dos meses do ano.

Tabela 4 – Filtros aplicados aos textos das consultas abstratas.

```
<filtrosDeConsulta>
        <dicionarios>
                <dicionario nome= "termos" >
                        <termo> Universidade do Rio de Janeiro </termo>
                        <termo> UFRJ </termo>
                        <termo> Departamento de Ciência da Computação </termo>
                        <termo> DCC </termo>
                </dicionario>
                <dicionario nome= "cidade" >
                        <termo> Rio de Janeiro </termo>
                        <termo> São Paulo </termo>
                        <termo> Guanambi </termo>
                        <termo> Guarani </termo>
                </dicionario>
```

```
            <dicionario nome= “stopwords” >
                <importar dicionario = “PORTUGUESE_STOPWORDS” >
            </dicionario>
            <dicionario nome= “mes” >
                <importar dicionario = “PORTUGUESE_MONTHS” >
            </dicionario>
        </dicionarios>
        <filtro>
            [\{termos\}\{stopwords\}]*
        </filtro>
        <filtro>
            \{cidade}, [0-9]\{1,2\} de \{mes\} de [0-9]\{4\}
        </filtro>
        <filtro>
            ^(Aluno: ) [a-zA-Z\n]+
        </filtro>
</filtrosDeConsulta>
```

A partir dos filtros de consultas definidos na Tabela 4, as consultas de identificadores iguais a 1, 2, 3 ou 4 serão desconsideradas da análise. Como as consultas em inglês foram geradas a partir das consultas em português, as consultas 13, 14, 15 e 16 também serão desconsideradas.

4.2.3.2 Execução de consultas

Nesse teste de mesa foi utilizada uma única fonte de documentos, o Google. Para tal, cada uma das consultas geradas foi executada no buscador Google, porém apenas os dez resultados mais relevantes foram considerados, o que engloba todos os resultados exibidos na primeira página da ferramenta. A

Tabela 5 e a Tabela 6 apresentam o resultados encontrados na execução de duas das consultas listadas na Tabela 3.

Tabela 5 – Resultado da execução da primeira consulta através do buscador Google.

Consulta:

A lógica difusa ou lógica fuzzy é uma extensão da lógica booleana que admite valores lógicos intermediários entre o FALSO (0) e o VERDADEIRO (1); por exemplo o valor médio 'TALVEZ' (0,5). Isto significa que um valor lógico difuso é um valor qualquer no intervalo de valores entre 0 e 1. Este tipo de lógica engloba de certa forma conceitos estatísticos principalmente na área de Inferência.

Resultados:

http://www.facebook.com/pages/L%C3%B3gica-difusa/104106919626684

http://www.facebook.com/pages/L%C3%B3gica-Fuzzy/111910592161021

http://pt-br.facebook.com/pages/L%C3%B3gica-difusa/104106919626684?sk=info

http://pt.wikipedia.org/wiki/L%C3%B3gica_difusa

http://pt.wikiteka.com/documento/prova-12/

http://api.adm.br/evalforum/?page_id=196

http://www.proz.com/kudoz/english_to_portuguese/computers%3A_software/3235487-fuzzy_logic.html

http://peteletricaunifei.wordpress.com/pesquisas/silas-toledo/

http://www.producao.ufrgs.br/arquivos/disciplinas/398_aula_2_prod_2.ppt

http://www.tiosam.org/?q=L%C3%B3gica_difusa

Tabela 6 – Resultado da execução de consulta em inglês através do buscador Google.

Consulta:

The fuzzy logic began with the 1965 draft, the theory of fuzzy sets by Lotfi Zadeh. Although fuzzy logic has been applied to many fields, from the control theory of artificial intelligence, remains controversial among most statisticians, who prefer the Bayesian logic, and some control engineers, who prefer the traditional bivalent logic.

Resultados:

http://en.wikipedia.org/wiki/Fuzzy_logic

http://www.iwawaterwiki.org/xwiki/bin/view/Articles/Riskanalysisfuzzylogicandmanagementofuncertainty?viewer=code&showlinenumbers=0

http://www.iwawaterwiki.org/xwiki/bin/view/Articles/Riskanalysisfuzzylogicandmanagementofuncertainty?viewer=code

http://www.iwawaterwiki.org/xwiki/bin/view/Articles/Riskanalysisfuzzylogicandmanagementofuncertainty

http://fuzzy-logic.co.tv/

www.scribd.com/doc/51204677/13/Fuzzy-logic

http://www.intpforum.com/showthread.php?p=222639

http://wn.com/fuzzy_logic

http://bongo-www.thefullwiki.org/Fuzzy_logic

http://traders-library.com/download/Lotfi%20Zadeh%20,%20King-Sun%20Fu%20-%20Fuzzy%20Sets%20and%20Their%20Applications%20to%20Cognitive%20and%20Decision%20Processes.pdf

Os resultados encontrados em todas as consultas serão comparados ao documento analisado na fase seguinte.

É importante observar que o Google pode vir a retornar URL's nas quais o documento analisado esteja disponível. Para evitar que downloads e comparações desnecessárias sejam efetuados, ao submeter a análise, o usuário da ferramenta deve poder submeter também uma lista de URL's, para quando o buscador utilizado for o Google, ou uma outra forma de representação de caminho para o documento retornado, quando o buscador utilizado for outro, como, por exemplo, um ID quando a base de documentos for um SGBD. Todos os documentos que coincidirem com essa lista deixarão de ser analisados.

Quando um mesmo documento for retornado por mais de uma consulta, esse será analisado apenas uma vez, porém a informação com o número de vezes que esse documento for citado pelos algoritmos de recuperação de documentos será armazenado para análises posteriores.

4.2.3.3 Comparação de documentos

Todos os métodos de comparação devem retornar um conjunto de similaridades entre dois documentos, onde cada similaridade é representada pela tupla: similaridade, valorada entre 0 e 1, trecho do primeiro documento e trecho do segundo documento. Nesse teste mesa, foi utilizado um único método de comparação entre documentos foi utilizado.

O método utilizado é simples, e, basicamente, retorna um conjunto de similaridades entre dois documentos, sendo cada similaridade resultante da comparação entre dois parágrafos, um de cada documento. Todos os parágrafos do primeiro documento (doc1) são comparados com os parágrafos do segundo documento (doc2).

No método aplicado, o resultado da comparação entre dois parágrafos será 1, caso os parágrafos sejam iguais ou um parágrafo contenha o outro, ou um valor designado pela fórmula abaixo:

$$sim(p1, p2) = \frac{|\mathrm{W}(p1) \cap \mathrm{W}(p2)|}{|\mathrm{W}(p1) \cup \mathrm{W}(p2)|},$$

onde W(*p*) representa o conjunto de palavras presentes no parágrafo *p*. De acordo com a fórmula, a similaridade entre os parágrafos *p1* e *p2*, *sim*(*p1*,*p2*), é o percentual de palavras em comum entre os dois parágrafos, no universo de palavras que os compõem.

O método também possui recurso para verificar se os documentos comparados estão em idiomas distintos, nesse teste de mesa os idiomas são limitados a português e inglês, e caso estejam, os textos são traduzidos para o inglês com o auxílio do Google Tradutor.

A Tabela 7 apresenta exemplos de similaridades encontradas entre o documento analisado e alguns dos documentos encontrados. Na tabela é possível verificar que, através do método utilizado, parágrafos copiados inteiramente retornam o valor 1, no entanto, parágrafos parafraseados ou que tiveram muitas de suas palavras alteradas por sinônimos tendem a ter baixa similaridade por esse método, como é o caso do parágrafo parafraseado que obteve similaridade de 0,25. Nesse caso, podem ser testados métodos que levem em consideração sinônimos ou mesmo que façam comparações semânticas nos textos.

Muitos são os métodos que podem ser utilizados nessa fase. Nesta dissertação, acredita-se que um bom método seja aquele que tenha um alto índice de recuperação dos documentos fontes de plágio, porém mantendo um índice de precisão aceitável. A seção 5.3 testa diversos desses métodos e apresenta o índice de retorno e a precisão encontrados em cada um.

Tabela 7 – Exemplos de similaridades encontradas.

ID	*p1* (Documento Analisado)	*p2* (http://pt.wikipedia.org/wiki/L%C3%B3gica_difusa)	Similaridade
1	A lógica difusa ou lógica fuzzy é uma extensão da lógica booleana que admite valores lógicos intermediários entre o FALSO (0) e o VERDADEIRO (1); por exemplo o valor médio 'TALVEZ' (0,5). Isto significa que um valor lógico difuso é um valor qualquer no intervalo de valores entre 0 e 1. Este tipo de lógica engloba de certa forma conceitos estatísticos principalmente na área de Inferência.	A lógica difusa ou lógica fuzzy é uma extensão da lógica booleana que admite valores lógicos intermediários entre o FALSO (0) e o VERDADEIRO (1); por exemplo o valor médio 'TALVEZ' (0,5). Isto significa que um valor lógico difuso é um valor qualquer no intervalo de valores entre 0 e 1. Este tipo de lógica engloba de certa forma conceitos estatísticos principalmente na área de Inferência.	**1,0**
2	A lógica difusa deve ser vista mais como uma área de pesquisa sobre tratamento de incerteza, ou como um conjunto de modelos matemáticos dedicados ao tratamento da incerteza, do que como uma lógica. A lógica difusa é normalmente associada ao uso de uma teoria proposta por Lukasiewicz, a teroria de conjuntos difusos.	A lógica fuzzy deve ser vista mais como uma área de pesquisa sobre tratamento da incerteza, ou uma família de modelos matemáticos dedicados ao tratamento da incerteza, do que uma lógica propriamente dita. A lógica difusa normalmente está associada ao uso da teoria de conjuntos fuzzy proposto por Lukasiewicz.	**0,64**
3	As implementações da lógica difusa permitem que estados indeterminados possam ser tratados por ferramentas de controle. Dessa forma, é possível avaliar conceitos não-quantificáveis, como, por exemplo, avaliar a temperatura (quente, morno, médio, etc...) e o sentimento de felicidade (radiante, feliz, apático, triste...)	As implementações da lógica difusa permitem que estados indeterminados possam ser tratados por dispositivos de controle. Desse modo, é possível avaliar conceitos não-quantificáveis. Casos práticos: avaliar a temperatura (quente, morno, médio, etc...), o sentimento de felicidade(radiante, feliz, apático, triste...), a veracidade de um argumento (correctíssimo, correcto, contra-argumentativo, incoerente, falso, totalmente erróneo, etc..)	**0,47**
4	Muitos pesquisadores de versões booleanas de lógica	Muitos pesquisadores de versões booleanas de lógica não aceitam a	**0,51**

	não aceitam a lógica difusa como uma verdadeira lógica, no sentido em que aceitam, por exemplo, a lógica modal.	lógica fuzzy como uma verdadeira lógica, no sentido em que aceitam, por exemplo, a lógica modal. Isso pode ser associado a diferentes fatos, entre eles o fato de muitos modelos permitirem soluções aproximadas que não correspondem a uma "verdade" lógica.	
5	Essa não aceitação pode ser associada a diferentes fatos, entre eles o fato de muitos modelos permitirem soluções aproximadas que não correspondem a uma "verdade" lógica.	Muitos pesquisadores de versões booleanas de lógica não aceitam a lógica fuzzy como uma verdadeira lógica, no sentido em que aceitam, por exemplo, a lógica modal. Isso pode ser associado a diferentes fatos, entre eles o fato de muitos modelos permitirem soluções aproximadas que não correspondem a uma "verdade" lógica.	**0,49**
6	A lógica booleana é normalmente referenciada como lógica nítida, ao se trabalhar com lógica difusa.	Ao trabalhar com a lógica fuzzy é comum chamar a lógica booleana de lógica nítida.	**0,25**
	***p1* (Documento Analisado)**	***p2* (http://en.wikipedia.org/wiki/Fuzzy_logic)**	**Similaridade**
7	The fuzzy logic began with the 1965 draft, the theory of fuzzy sets by Lotfi Zadeh. Although fuzzy logic has been applied to many fields, from the control theory of artificial intelligence, remains controversial among most statisticians, who prefer the Bayesian logic, and some control engineers, who prefer the traditional bivalent logic.	Fuzzy logic began with the 1965 proposal of fuzzy set theory by Lotfi Zadeh.[2][3] Though fuzzy logic has been applied to many fields, from control theory to artificial intelligence, it still remains controversial among most statisticians, who prefer Bayesian logic, and some control engineers, who prefer traditional two-valued logic	**0,69**

As similaridades 0,51 e 0,49 foram dadas a textos que fazem parte de um mesmo parágrafo, mas que devido a uma pequena mudança em cada texto, fez com que os mesmos não estivessem mais completamente contidos no parágrafo original. A técnica de comparação de textos utilizada é sensível a essa técnica de plágio e pode ser facilmente burlada. No entanto, caso o texto seja reestruturado, uma nova análise pode ser feita e novos valores podem ser encontrados.

Uma reavaliação da estrutura previamente definida para o documento pode ser útil para suprir falhas na definição, feita pela ferramenta, de início e de fim de parágrafos ou seções. Formatos de arquivo como o PDF podem tornar difícil a identificação de parágrafos. Com uma reestruturação do documento, é possível reverter trechos de parágrafos que aparecem em páginas diferentes, por exemplo, em um único parágrafo.

A ferramenta proposta permite o uso de várias técnicas para a reestruturação de documentos. Nesse teste de mesa foi usada uma técnica simples, na qual dois ou mais parágrafos seqüenciais, que tenham alguma similaridade para com um mesmo parágrafo, sejam unidos e novamente comparados. Utilizando essa técnica, é encontrada a similaridade apresentada na Tabela 8.

Tabela 8 – Similaridade encontrada após reestruturação do documento.

ID	***p1* (Documento Analisado)**	***p2* (http://pt.wikipedia.org/wiki/L%C3%B3gica_difusa)**	**Similaridade**
8	Muitos pesquisadores de versões booleanas de lógica não aceitam a lógica difusa como uma verdadeira lógica, no sentido em que aceitam, por exemplo, a lógica modal. Essa não aceitação pode ser associada a diferentes fatos, entre eles o fato de muitos modelos permitirem soluções aproximadas que não correspondem a uma "verdade" lógica.	Muitos pesquisadores de versões booleanas de lógica não aceitam a lógica fuzzy como uma verdadeira lógica, no sentido em que aceitam, por exemplo, a lógica modal. Isso pode ser associado a diferentes fatos, entre eles o fato de muitos modelos permitirem soluções aproximadas que não correspondem a uma "verdade" lógica.	**0,84**

A similaridade 0,84 demonstra uma alta probabilidade dos dois parágrafos terem sido plagiados de um único parágrafo.

4.2.3.4 Análise de plágio

Nessa fase, o analista de plágio deve visualizar os documentos e similaridades encontrados, selecionar os mais relevantes e descartar similaridades que não apresentem plágio.

Com o objetivo de filtrar as similaridades, o analista deve ter a sua disposição funcionalidades como filtro de valor mínimo e máximo de similaridade, filtro de

similaridades que tenham sido referenciados somente pelo documento e/ou pelo parágrafo, filtro que descarte similaridades sinalizadas, pelo analista, como não relevantes, entre outros filtros que facilitem a visualização das similaridades.

Nessa fase, o analista poderia descartar similaridades irrelevantes, como a apresentada na Tabela 9, e destacar positivamente similaridades que sejam realmente plágio, como as de identificadores 1, 2, 4 e 8.

Tabela 9 – Exemplo de similaridade irrelevante.

p1 **(Documento Analisado)**	***p2*** **(http:// geocities.ws/thiago_ferauche/projetos/Fuzzy.doc)**	**Similaridade**
Trabalho sobre Lógica Fuzzy	Trabalho de Lógica Fuzzy	**0,6**

O trabalho do analista de plágio deve ser cuidadoso, porém, pode ser inviável uma análise cuidadosa de cada similaridade encontrada. Por esse motivo, um analista deve focar em um grupo seleto de similaridades. Esse grupo pode ser de similaridades com índices maiores do que 0,5, por exemplo. Caso o índice mínimo seja de 0,5, a similaridade de índice 0,25 apresentada na Tabela 7, embora seja um plágio, não será detectado como tal.

4.2.3.5 Relatórios

Nessa fase são gerados relatórios para avaliação final do documento. Esses relatórios devem exibir índices de plágio, documentos similares ao analisado e outros dados e formas de apresentação que facilitem a análise de usuário. Mais adiante, na seção 5.2.5, são exibidos exemplos de relatórios na Figura 26 e na Figura 27.

5 Validação da arquitetura

De modo a validar a arquitetura proposta, foi implementado para testes um protótipo, em forma de software, batizado de Textual Plagiarism Analyzer, ou simplesmente TEXPLAN. Este capítulo detalha a implementação, discutindo a tecnologia utilizada, os objetivos alcançados e as limitações encontradas. O capítulo também apresenta o experimento utilizado como teste para a ferramenta, detalhando os procedimentos adotados e seus resultados.

5.1 Especificação da ferramenta

5.1.1 Interações com o usuário

A interação com a ferramenta é feita em sua grande parte por um perfil de acesso chamado de Analista de Plágio, sendo esse o responsável por todo o processo de detecção de plágio.

O Analista de Plágio deve ter conhecimentos básicos sobre plágio, entendendo o conceito e as formas nas quais pode se apresentar. Entre as atribuições do perfil em questão estão: submeter um documento, especificar e requisitar uma nova análise, verificar o andamento da análise, verificar similaridades encontradas e julgar as mais relevantes e extrair da ferramenta relatórios resultantes da análise de plágio.

A Figura 15 exibe o diagrama de casos de uso da ferramenta, evidenciando as principais interações.

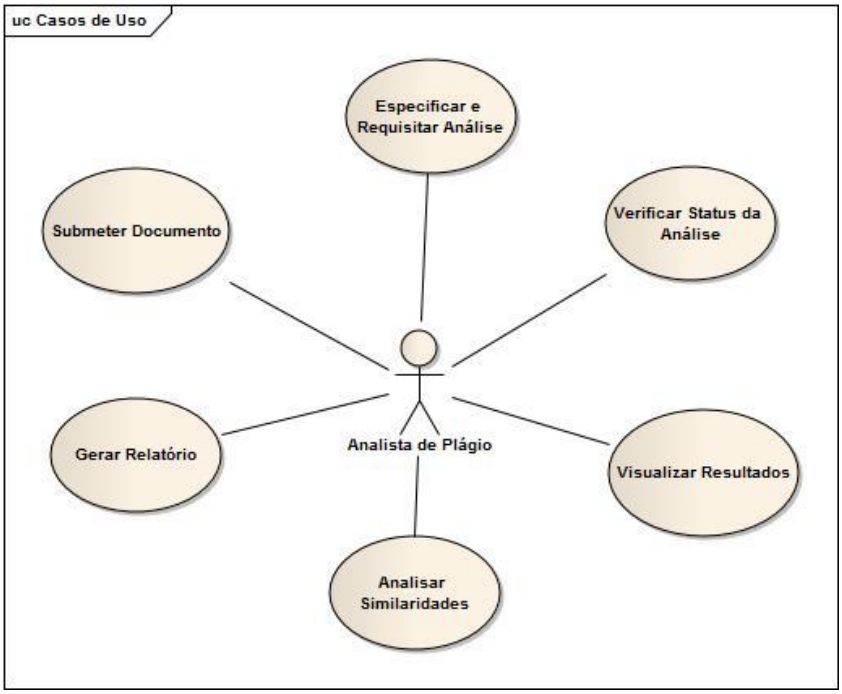


Figura 15 - Diagrama de casos de uso.

5.1.2 Processo de análise

O fluxo principal da ferramenta pode ser visto na Figura 16, que, através de um diagrama de atividades, descreve todo processo de detecção de plágio ao qual a ferramenta dá suporte.

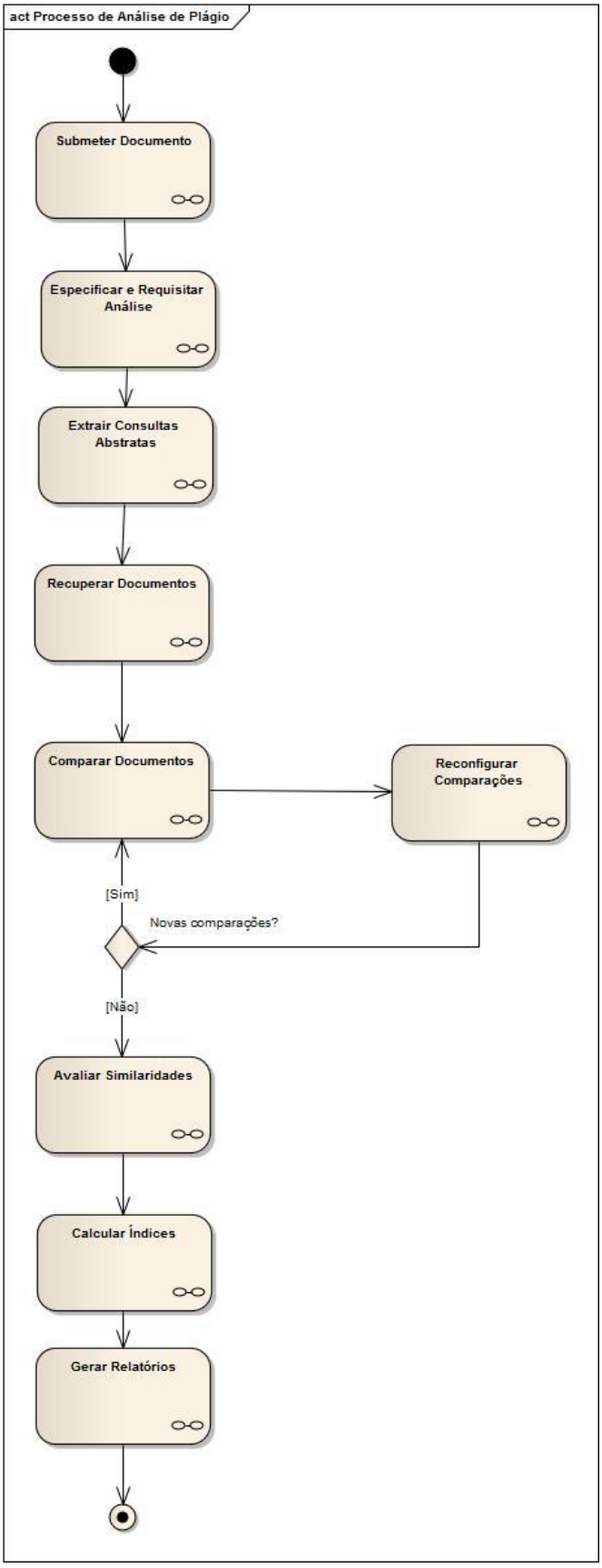


Figura 16 - Processo de análise de plágio.

O processo descrito na Figura 16 não é executado em um único passo, necessitando de diversas interações com o usuário. Essas iterações foram explicitadas no diagrama de caso de uso apresentado (Figura 15). Dentre os casos de uso existentes, dois merecem destaque: especificar e requisitar análise e analisar similaridades.

Ao requisitar uma análise de plágio, é necessário que o Analista de Plágio especifique os algoritmos que serão utilizados nas diferentes fases da análise, como demonstra o diagrama de atividades na Figura 17. Esses algoritmos devem ser previamente cadastrados na ferramenta pelo Pesquisador de Algoritmos.

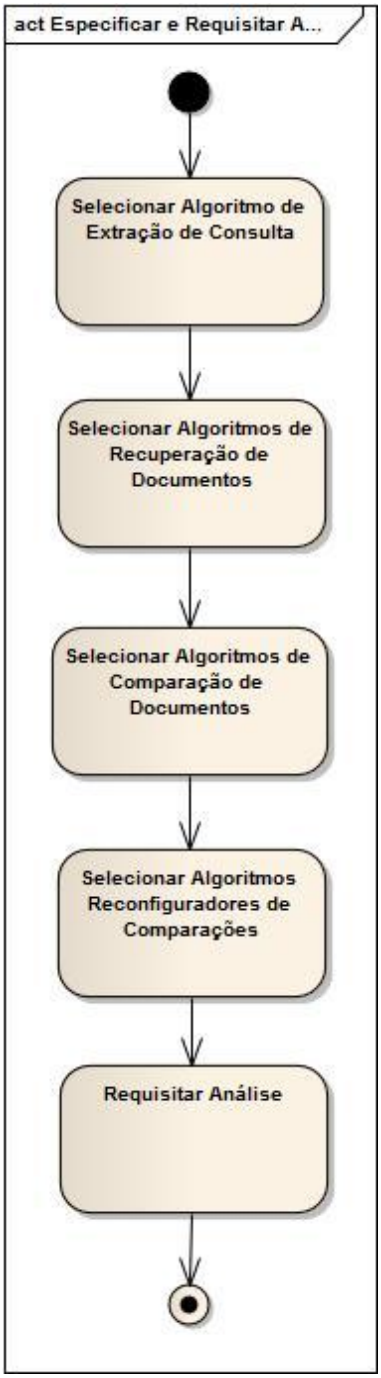


Figura 17 – Processo de especificar e requisitar análise de plágio.

No caso de uso "avaliar similaridades", o Analista de Plágio deve definir filtros que serão utilizados pela ferramenta para apresentar ao mesmo as similaridades previamente encontradas. Cada similaridade resultante pode ser então avaliada pelo analista. Esse processo está definido na Figura 18.

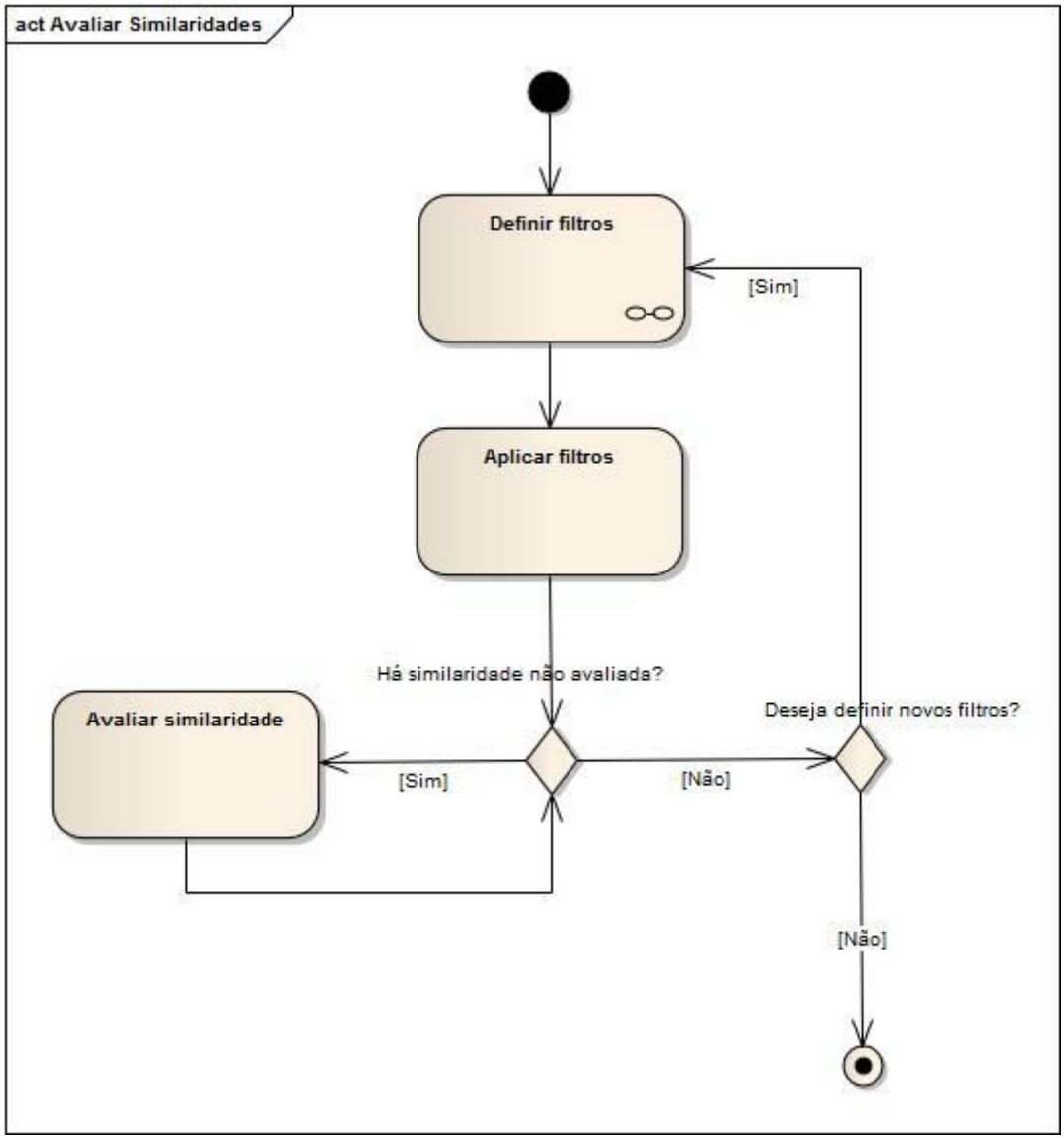


Figura 18 – Processo de avaliar similaridades.

5.2 Implementação do protótipo

O capítulo 3 apresenta diversas linhas de pesquisa, que produzem algoritmos que podem vir a ser testados, na ferramenta aqui proposta, como parte de uma solução para o auxílio a detecção de plágio. No entanto, muitos desses algoritmos demandam grande poder computacional, bem como a recuperação de documentos pode demandar grande capacidade de transmissão de dados.

No intuito de oferecer uma melhor infra-estrutura para a execução dos algoritmos citados acima, a implementação visou tirar proveito do paralelismo oferecido pelas tecnologias atuais para melhorar o tempo de resposta das análises de plágio. O paralelismo no processamento das diversas fases da análise se mostra muito útil, para que, por exemplo, enquanto uma fase que consome grande largura de banda e pouco

processamento estiver sendo executada, outra fase que consuma mais processamento também possa ser executada.

A ferramenta é assíncrona, não limitando o tempo necessário ao processamento dos algoritmos, e as operações utilizadas são transações bem delimitadas, permitindo que a qualquer momento os resultados parciais de uma análise sejam visualizados e/ou a análise seja interrompida, caso os resultados até então obtidos sejam satisfatórios.

5.2.1 Tecnologia utilizada

A linguagem escolhida para implementar a ferramenta foi a linguagem JAVA[14], por ser amplamente difundida, com farta quantidade de soluções para problemas rotineiros e também problemas científicos, disponíveis na Web. Na seleção pela linguagem, também pesou o fato da linguagem ser independente de sistema operacional, o que permitiu a execução da ferramenta em diferentes ambientes, como Windows, Linux e MacOS.

Para tirar proveito não só do paralelismo de execução de processos em uma única máquina, mas também tornar viável o uso de inúmeras máquinas ou, até mesmo, do processamento em nuvem, foi adotada uma solução na qual as tarefas são dividas, de forma independente, e, utilizando-se de filas de processamento, disponibilizadas para processamento. Dessa forma, tarefas de uma mesma análise podem ser processadas por diferentes máquinas, independente de hardware, sistema operacional ou localização física.

A solução utilizada para o problema de filas de processamento foi a disponibilizada pela própria criadora da linguagem Java. A API Java Message Service (JMS) define um padrão para a distribuição de mensagens em aplicações corporativas, permitindo, segundo a Oracle Corporation (2011c), processamento assíncrono das mensagens e até mesmo compartilhamento de transação entre o destinatário e o remetente da mensagem.

A JMS permite a criação de uma fila para a qual é possível enviar mensagens solicitando o processamento de uma tarefa. Por outro lado, também é possível cadastrar consumidores, que são programas capazes de processar uma tarefa colocada na fila. A Figura 19 representa a iterações de dois programas diferentes (client 1 e client 2) com a fila (queue) JMS. Toda a interação é feita através de mensagens.

[14] Mais informações sobre a linguagem JAVA podem ser encontradas no site http://www.oracle.com/us/technologies/java/index.html

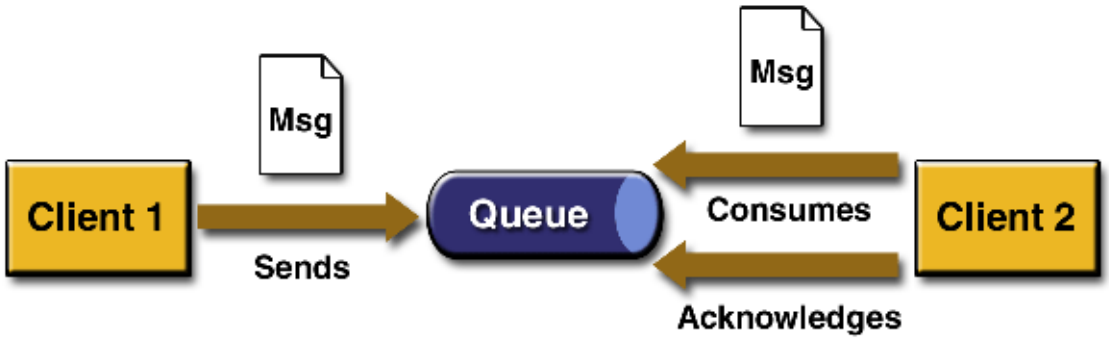


Figura 19 - Mensagem ponto-a-ponto (SUN MICROSYSTEMS, INC., 2002).

Por incluir facilidades para o uso de transações e da API JMS, foi escolhido utilizar a tecnologia Enterprise Java Beans (EJB). O EJB implementa diversos padrões para o desenvolvimento de sistemas, bem como possui facilidades para o uso de JMS, como configuração simples para criação de fila, instanciação automática de consumidores, e cadastramento dos mesmos na fila, e controle das transação no processamento das tarefas. O servidor de aplicações adotado para uso do EJB nessa implementação foi o JBoss AS 5.

Para armazenar os dados e documentos, com os quais a ferramenta trabalha, foi utilizado o SGBD[15] MySQL 5.5, com o uso da API Hibernate para mapeamento das classes de domínio, controle de transação a nível de domínio e diminuição da dependência do SGBD utilizado.

Para viabilizar o uso da ferramenta, foi desenvolvida uma interface web utilizando os arcabouços JavaServer Faces 1.2 e Richfaces 3.3.

5.2.2 Interfaces definidas

De modo a possibilitar a inserção de novos algoritmos e técnicas computacionais na ferramenta, foram definidas classes com métodos abstratos que deverão ser implementados, de modo a executar a técnica desejada. Essas classes são definidas no diagrama apresentado na Figura 20, e explicadas em seguida.

[15] Sistema Gerenciador de Banco de Dados.

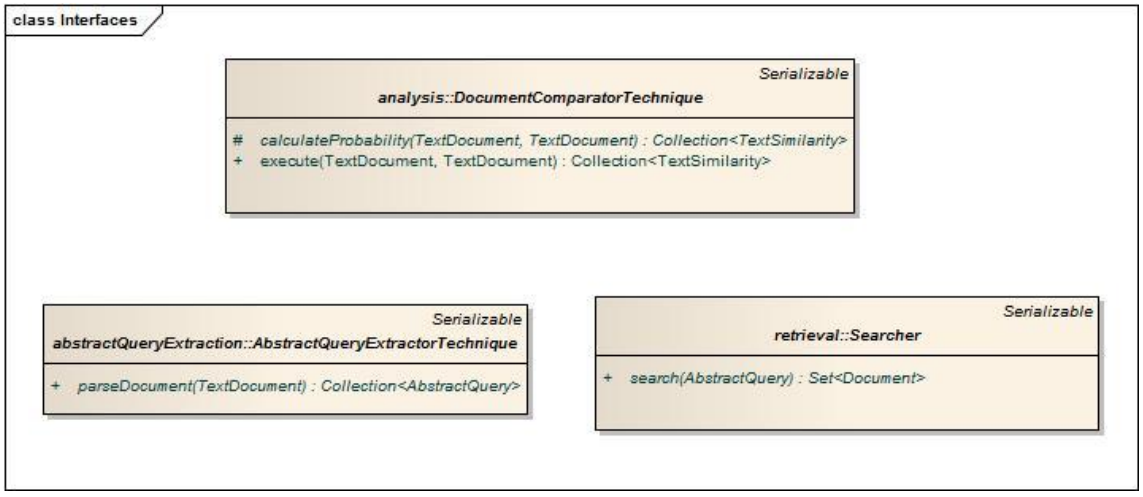


Figura 20 – Interfaces definidas para a implementação das técnicas utilizadas.

A classe abstrata *DocumentComparatorTechnique* é utilizada para a implementação dos ACD's. A classe possui o método abstrato *calculateProbability*, que recebe como parâmetros dois documentos textuais e retorna uma coleção de similaridades entre eles. Esse método deverá ser implementado utilizando o ACD.

A classe *AbstractQueryExtractorTechnique* possui o método abstrato *parseDocument*, que recebe como parâmetro o documento alvo da análise de plágio. A classe *AbstractQueryExtractorTechnique* deverá ser estendida e o método *parseDocument* deverá ser implementado com o AEC desejado.

A classe *Searcher* deve ser estendida, e o método *search* implementado, para tornar disponível à ferramenta um ARD. O método *search* recebe como parâmetro uma Consulta Abstrata e retorna uma coleção de documentos.

5.2.3 Implementações das interfaces

Para validar a arquitetura, foi necessária a implementação das interfaces definidas no protótipo. Na implementação das interfaces foram utilizados algoritmos e técnicas apresentados na seção 3 desta dissertação, ou variações do mesmo, conforme será apresentado nas subseções a seguir.

5.2.3.1 Implementações de AbstractQueryExtractorTechnique

Três implementações da interface AbstractQueryExtractorTechnique foram feitas. Todas as implementações geram Consultas Abstratas compostas apenas por um texto simples, ao invés de XML's complexos, pois as mesmas serão executas como consultas simples no Google.

A primeira implementação recupera todos os parágrafos contidos no documento analisado e cria, para cada parágrafo, uma Consulta Abstrata contendo apenas o texto do mesmo.

A segunda implementação recupera todas as sentenças do documento analisado e, através de uma simplificação do algoritmo de sumarização de Luhn (LUHN, 1958), cria um *ranking* das sentenças por ordem de importância. Apenas trinta por cento das sentenças, localizadas mais ao topo do *ranking* são utilizadas na criação de Consultas Abstratas, onde cada Consulta Abstrata contém apenas o texto simples de uma sentença.

A simplificação do algoritmo de Luhn utilizada calcula a relevância de uma sentença como a média das relevâncias dos seus termos, excluídos os *stop words* e aplicada radicalização. A relevância de um termo é calculada simplesmente pela sua frequência no documento.

A última implementação também recupera todas as sentenças do documento analisado, no entanto, diferente da implementação anterior, utiliza todas como Consultas Abstratas.

5.2.3.2 Implementação de Searcher

Foi feita uma única implementação da interface Searcher que utilizou a Internet como base de documentos e o Google como ferramenta de consulta a mesma. Para tal fim, foi utilizada a API JAVA *Search Engine Wrapper*[16] para execução de consultas no Google.

5.2.3.3 Implementações de DocumentComparatorTechnique

As implementações propostas definem similaridades entre duas sentenças de documentos, onde uma sentença é definida como parte do texto de um parágrafo, iniciado no início de um parágrafo ou após fim de outra sentença e finalizado por um ponto (“.”, “!” ou “?”) ou por uma sequência de pontos.

Um documento pode conter milhares de sentenças, o que torna uma comparação de todas as sentenças de um documento com as sentenças de outro inviável, em termos de tempo computacional, quando um grande número de documentos deve ser comparado. Para viabilizar a execução das comparações, o algoritmo lança mão do uso do MySQL Fulltext (GOLUBCHIK, 2003), descrito na seção 3.5.

[16] Disponível em http://wing.comp.nus.edu.sg/~tanyeefa/downloads/searchenginewrapper/

Cada sentença do documento suspeito de plágio é comparada com as cinco sentenças mais similares no possível documento fonte, segundo o MySQL Fulltext . O que reduz a complexidade da comparação entre dois documentos de O(*nm*) para O(*n*), onde *n* é o total de sentenças no documento suspeito e *m* é o total de sentenças no documento comparado.

Na validação da arquitetura foi utilizada a língua inglesa, tanto para os documentos suspeitos, quanto para os documentos comparados, porém, durante o desenvolvimento do algoritmo, foi dado suporte a comparação de sentenças em língua inglesa com sentenças em língua portuguesa. Nesse caso, no momento do uso do MySQL Fulltext a sentença do documento suspeito deve ser traduzida para a língua do documento comparado.

No momento da comparação entre as sentenças, caso seja necessário, a sentença pertencente ao documento comparado é traduzida para a língua do documento suspeito. A tradução é feita a partir da língua do documento comparado para a língua do documento suspeito, e não o caminho inverso, por se tratar do caminho naturalmente seguido na criação do plágio.

Toda tradução é feita utilizando a ferramenta Google Tradutor.

Todos os passos descritos acima foram utilizados nas implementações descritas nas subseções a seguir. Todas as implementações desconsideraram valores de similaridades menores que 0,5.

5.2.3.3.1 Levenshtein de sentenças

Nessa implementação a similaridade entre as sentenças é representada por um valor de entre zero e um, estipulado por uma variação do algoritmo de Levenshtein. Nessa variação, a distância entre duas sentenças é definida como o número de trocas, inserções ou exclusões de palavras necessárias para se transformar uma sentença em outra.

A distância máxima entre duas sentenças será o maior entre as quantidades de palavras em cada sentença. Dessa forma, a seguinte formula foi utilizada para definir a similaridade entre duas sentenças:

$$sim(s1, s2) = 1 - \frac{leven(s1, s2)}{\max(\ |words(s1)|,\ |words(s2)|\ )}$$

, onde $leven(s1, s2)$ é a distância de Levenshtein entre as sentenças *s1* e *s2*, e $words(s)$ representa o conjunto de palavras da sentença s.

5.2.3.3.2 Levenshtein de sentenças por token sem stopwords

Nessa implementação, a similaridade entre duas sentenças também é calculada a partir de uma variação do algoritmo de Levenshtein. No entanto, as sentenças comparadas são pré-processamento, tendo os seus *stop words* retirados e os demais termos reduzidos ao seu radical.

Como *stop words* da língua inglesa foi considerado o conjunto de palavras disponibilizado pela API JAVA do Apache Lucene[17]. Para redução das demais palavras em língua inglesa aos seus radicais foi utilizado o algoritmo de *stemming* (vide seção 3.3.2) de Porter, tendo sido utilizada a implementação do algoritmo em JAVA disponibilizada pelo próprio Martin Porter em seu *website*[18].

Logo, a similaridade entre duas sentenças *s1* e *s2* é definida pela fórmula:

$$sim(s1, s2) = 1 - \frac{leven(\ preProc(s1),\ preProc(s2))}{\max(\ |words(\ preProc(s1)\ )|,\ |words(\ preProc(s2)\ )|\ )}$$

, onde $preProc(s)$ é o texto resultante do pré-processamento da sentença *s*.

5.2.3.3.3 Bag of words por token sem stop words

Outra abordagem utilizada na comparação entre sentenças foi a de *bag of words* (definida na seção 3.4). Nessa abordagem, a ordem das palavras em uma sentença não é importante, o que facilita a detecção de plágio do tipo "alteração gramatical", como mudança de voz ativa para passiva, por exemplo.

Um exemplo de mudança gramatical pode ser visto nas sentenças "grandpa ate the apple" e "the apple was eaten by grandpa". Segundo a variação do algoritmo de Levenshtein apresentada na seção 5.2.3.3.2, a similaridade entre essas duas sentenças é zero, e será desconsiderado pela ferramenta. Esse exemplo mostra uma fragilidade do algoritmo anterior que pode ser suprida pelo uso de *bag of words*.

[17] Disponível em http://lucene.apache.org/
[18] Implementações do Porter Stemmer em http://tartarus.org/~martin/PorterStemmer/

Nessa implementação as sentenças são pré-processadas com o uso dos *stop words* do Apache Lucene e do algoritmo de *stemming* de Porter e disponibilizadas como um mapa onde cada entrada é um *token* e o número de vezes que o mesmo aparece na sentença. Ao processarmos as duas sentenças "grandpa ate the apple" e "the apple was eaten by grandpa" temos o *bag of words* como mostra a Tabela 10.

Tabela 10 – Matriz *bag of words com tokens.*

	grandpa	ate	appl	eaten
"grandpa ate the apple"	1	1	1	0
"the apple was eaten by grandpa"	1	0	1	1

A distância entre duas sentenças é então definida como a soma das diferenças de frequência de cada *token*, e a similaridade entre as mesmas é definida como:

$$sim(s1, s2) = 1 - \frac{distancia(s1, s2)}{distanciaMaxima(s1, s2)}$$

A distância máxima entre duas sentenças será obtida quando não houver tokens em comum entre elas. Dessa forma, a distância máxima pode ser definida como a soma das frequências de cada token nas duas sentenças:

$$distanciaMaxima(s1, s2) = \sum_{a \in tokens(s1,s2)} freqToken(a, s1) + freqToken(a, s2)$$

, onde $tokens(s1, s2)$ é o conjunto de *tokens* resultantes do pré-processamento das sentenças *s1* e *s2*, e $freqToken(a, s)$ é frequência com que um *token a* aparece em uma sentença *s*.

Sendo assim, a similaridade entre duas sentenças *s1* e *s2* é definida pela seguinte fórmula:

$$sim(s1, s2) = 1 - \frac{\sum_{a \in tokens(s1,s2)} |freqToken(a, s1) - freqToken(a, s2)|}{\sum_{a \in tokens(s1,s2)} freqToken(a, s1) + freqToken(a, s2)}$$

Respeitando a fórmula acima, a similaridade encontrada no exemplo da Tabela 10 é 0,66, o que supera o resultado do algoritmo de Levenshtein e destaca um possível plágio.

5.2.3.3.4 *Bag of words por synset sem* stop words

Nessa implementação também foi utilizada *bag of words*, no entanto, não foi utilizado *stemming*, e no seu lugar foram utilizados *synsets* (vide seção 3.6). Nesse caso, foram consideradas como iguais, as palavras que compartilhem de um mesmo *synset*.

Ao analisarmos o exemplo da subseção anterior, acrescido de uma mudança por sinônimo da palavra *grandpa* pela palavra *grandfather* na segunda sentença, a similaridade encontrada pelo algoritmo *bag of words* com *tokens* cai para 0,33, o que seria desconsiderado por este experimento. No entanto, se utilizarmos uma abordagem com sinônimos as palavras *grandpa* e *grandfather* seriam consideradas iguais.

No algoritmo anterior, ao encontrarmos duas palavras iguais, poderíamos simplificar a matriz *bag of words* mantendo apenas uma palavra e incrementando o número de aparições da mesma. Entretanto, ao utilizarmos sinônimos, duas palavras iguais podem estar presente em uma sentença com sentidos diferentes. A palavra *chave*, por exemplo, poderia aparecer em uma mesma sentença duas vezes, primeiramente referenciando o objeto chave e em seguida referenciando uma forma de se alcançar um objetivo, como no exemplo "ao abrir aquela porta secreta com esta chave, encontrará a chave do meu sucesso". Uma sentença similar seria "ao abrir aquela porta secreta com esta chave, encontrará o segredo do meu sucesso". Se considerarmos *segredo* e *chave* como a mesma palavra, poderíamos simplificar a matriz, no entanto, a palavra *segredo* pode estar associada a *secreto*.

Palavras podem ter diversos sentidos, e portanto diversos *synsets*. A palavra *key*, por exemplo, possui, segundo a WordNet[19] possui 22 sentidos, sendo 2 como adjetivo, 5 como verbo e 15 como nome. Sendo assim, determinar a qual palavra de uma sentença está associada uma palavra de outra sentença deixa de ser uma tarefa trivial.

[19] Consultada em 18 de agosto de 2011.

O intuito desse algoritmo é encontrar similaridades entre documentos que levantem suspeitas quanto a plágio. Logo, se compararmos duas sentenças, o algoritmo não peca por ser conservador e retornar o maior valor de similaridade possível. Foi considerado então que dadas duas sentenças *s1* e *s2* devem ser considerados os relacionamentos entre palavras que totalizem o maior número de associações possíveis entre *s1* e *s2*. Sendo assim, dados dois conjuntos de palavras *words1*, contendo as palavras de *s1*, e *words2*, contendo as palavras de *s2*, nos quais as palavras de *words1* possuem relacionamentos com uma ou mais palavras de *words2*, devemos eliminar os relacionamentos até que cada palavra em *words1* só esteja associada a uma única palavra em *words2*, e vice-versa. Essa eliminação dever ser feita de forma que o número de relacionamentos restantes seja o máximo possível.

O problema descrito no parágrafo anterior pode ser representado através de um grafo bipartido, como na Figura 21, e resolvido através de algoritmos aplicados aos mesmos. Na bibliografia de grafos pode-se associar o problema descrito ao problema do emparelhamento máximo em grafos bipartidos, o que pode ser resolvido através do uso do Algoritmo Húngaro (KUHN, 1955).

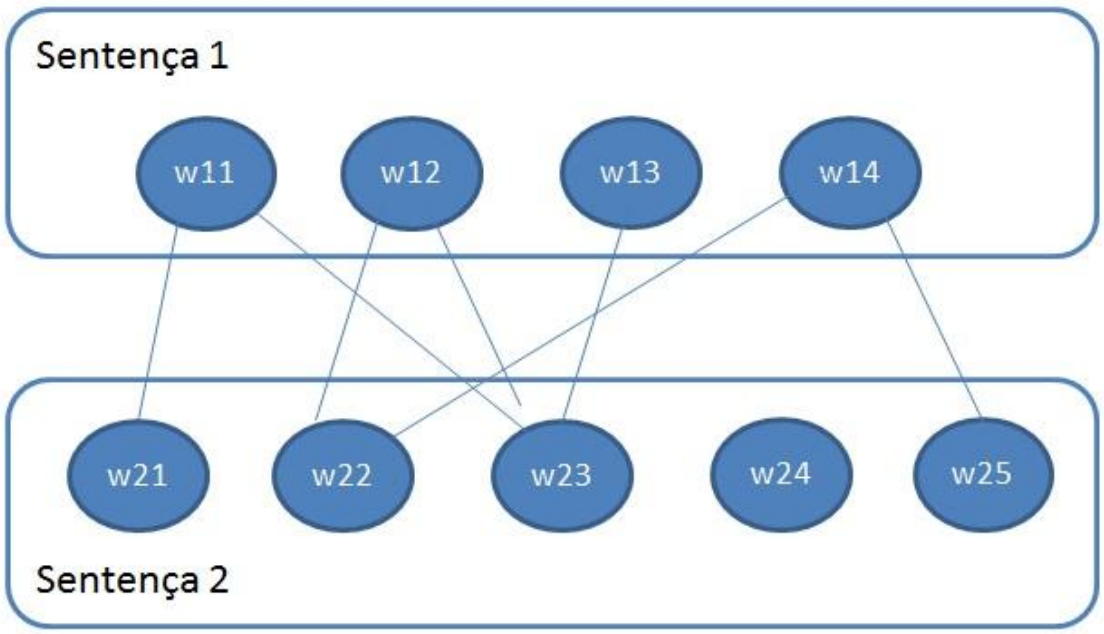


Figura 21 – Grafo bipartido representando as palavras das sentenças (vértices) e os seus relacionamentos de igualdade ou sinonímia (arestas).

Uma abordagem possível para execução do Algoritmo Húngaro é através do uso de matrizes. As arestas (relações de sinônimos) entre dois vértices (palavras) são

representadas com pesos, que nesta implementação teve o valor 1, quando as palavras pertencem a um mesmo *synset*, e 0, quando não há *synset* em comum. O algoritmo irá então determinar quais arestas devem ser mantidas para que a soma dos seus pesos seja máxima e cada vértice esteja ligado a uma única aresta.

Ao utilizarmos o Algoritmo Húngaro para comparação das sentenças "grandpa used the key to open the door and found the key to his success" e "the key was used by grandfather to open the door, that found the fundamental to his success", teremos a matriz apresentada na Tabela 11, após a exclusão de *stop words* e utilização da WordNet como fonte de *synsets*.

Tabela 11 - Matriz de relacionamentos entre palavras das sentenças *s1* e *s2*.

	key	used	grandfather	open	door	found	fundamental	his	success
grandpa	0	0	1	0	0	0	0	0	0
used	0	1	0	0	0	0	0	0	0
key	1	0	0	0	0	0	1	0	0
open	0	0	0	1	0	0	0	0	0
door	0	0	0	0	1	0	0	0	0
found	0	0	0	0	0	1	0	0	0
key	1	0	0	0	0	0	1	0	0
his	0	0	0	0	0	0	0	0	0
success	0	0	0	0	0	0	0	0	1

A matriz apresentada em Tabela 11 será analisada pelo Algoritmo Húngaro, que deverá retornar quais relações serão mantidas no emparelhamento máximo. O total de relações mantidas será utilizado no cálculo da similaridade entre as sentenças, conforme a fórmula abaixo:

$$sim(s1, s2) = \frac{emparelhamentoMaximo(preProcess\ (s1), preProcess\ (s2))}{maximo(\ |preProcess\ (s1)|,\ |preProcess\ (s2)|)}$$

, onde $emparelhamentoMaximo$ define o total de relações entre duas listas de palavras no seu emparelhamento máximo, e $preProcess(s)$ indica a lista de palavras de uma sentença *s* retirando-se suas *stop words*.

Para o exemplo da Tabela 11, a similaridade encontrada é 0,88. Já para o exemplo da subseção 5.2.3.3.3, o resultado encontrado é 1, que é um valor máximo e superior ao valor encontrado pelo algoritmo daquela subseção (0,66).

Nessa análise foi utilizada a implementação JAVA do Algoritmo Húngaro disponibilizada por Konstantinos Nedas em seu *website*[20], conforme licença disponível no mesmo.

5.2.4 Índice de plágio utilizado

Para validação da arquitetura foi definida a seguinte fórmula de calculo do índice de plágio para cada documento comparado ao documento suspeito:

Se maxSim(ds,dc) = 0,

$$Indice(ds, dc) = 0.$$

Senão,

$$Indice(ds, dc) = 0{,}5 * mediaSim(ds, dc) + 0{,}2 * maxSim(ds, dc) + 0{,}3 * repetiçõesResulBusca(dc)/maxRepetiçõesResulBusca(ds)$$

, onde *ds* é o documento suspeito e *dc* é o documento comparado, *mediaSim(ds,dc)* indica o valor médio das similaridades entre *ds* e *dc*, $maxSim(ds, dc)$ indica o maior valor entre as similaridades encontradas, $repetiçõesResulBusca(dc)$ indica o número de vezes que o documento *dc* foi retornado como resultado de uma busca durante a análise e $maxRepetiçõesResulBusca(ds)$ indica o número máximo de vezes que um documento se repetiu como resultados das buscas da análise.

5.2.5 Fluxo de telas e relatórios

O protótipo foi batizado de Textual Plagiarism Analyzer, ou simplesmente TEXPLAN. A tela inicial do sistema pode ser vista na Figura 22. O primeiro passo esperado do

[20] Implementação do Algoritmo Húngaro disponível em http://konstantinosnedas.com/dev/soft/munkres.htm.

usuário é a apresentação de seu login e senha, sendo possível criar um caso ainda não tenha. O login atualmente só é utilizado para identificar que avalia as similaridades encontradas.

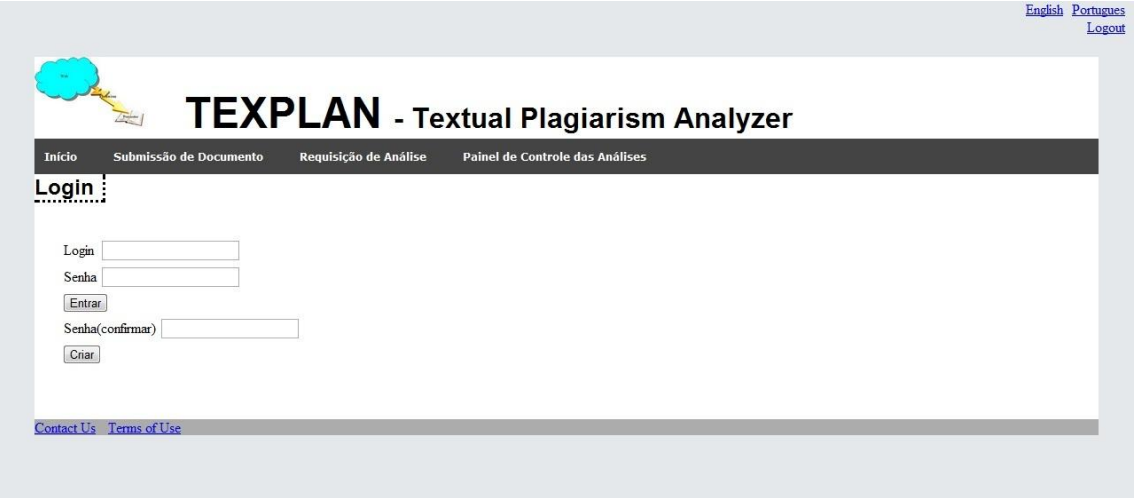


Figura 22 – Tela de login do sistema.

Antes de executar qualquer análise, é necessário cadastrar no sistema o documento que será analisado. Para isso, a tela de submissão de documento, apresentada na Figura 23, deve ser utilizada. Nessa tela é possível escolher entre três tipos de submissão: arquivo, URL e texto. A submissão de arquivo é justamente a apresentada na Figura 23. Os arquivos podem variar entre qualquer formato suportado pela API Apache TIKA[21], como doc, docx, pdf, ppt, txt e html.

[21] API JAVA utilizada neste trabalho para extração de textos em diferentes formatos. Disponível em http://tika.apache.org.

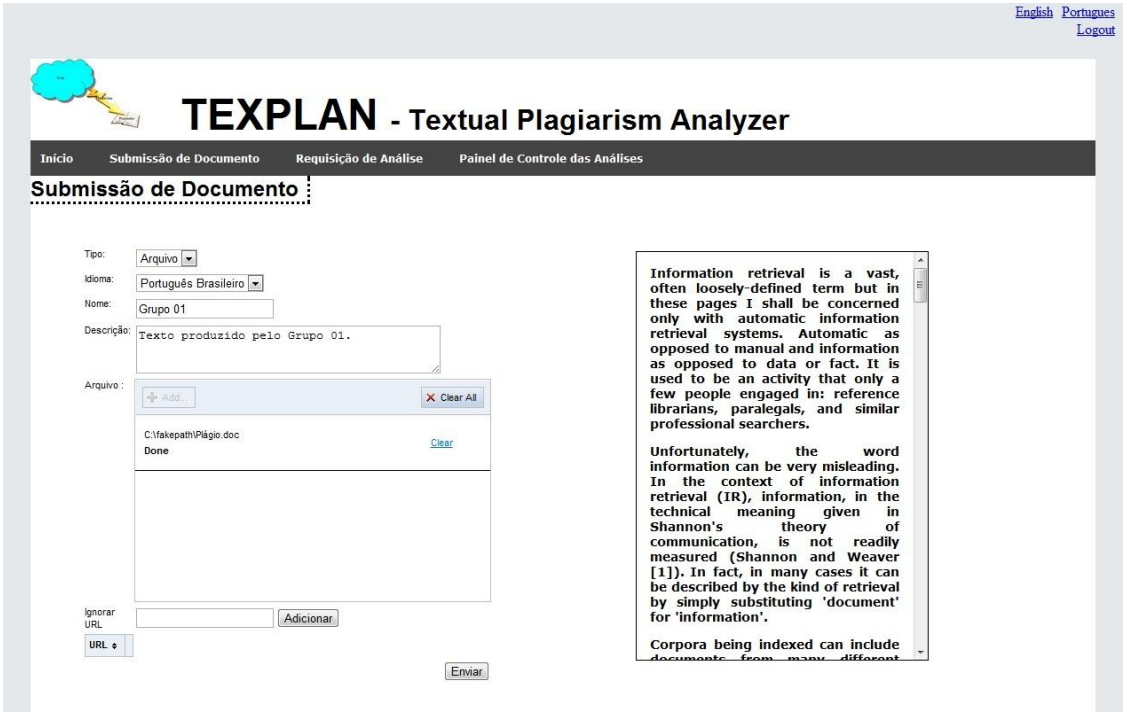


Figura 23 – Tela de submissão de documento.

A tela de submissão de documento também permite ao usuário indicar o idioma no qual o documento foi escrito e dar um nome e uma descrição ao mesmo. Além de permitir o cadastro URL's a serem desconsideradas na análise, conforme descrito na seção 4.2.1. tela também exibe ao usuário o texto extraído do documento, e que será considerado nas análises, já definidos os parágrafos, seções e sentenças.

A Figura 24 apresenta a tela de requisição de análise. Nessa tela o usuário dá um nome a análise, seleciona o documento a ser analisado, que será imediatamente apresentado na tela, e escolhe que algoritmos serão utilizados em cada fase da análise.

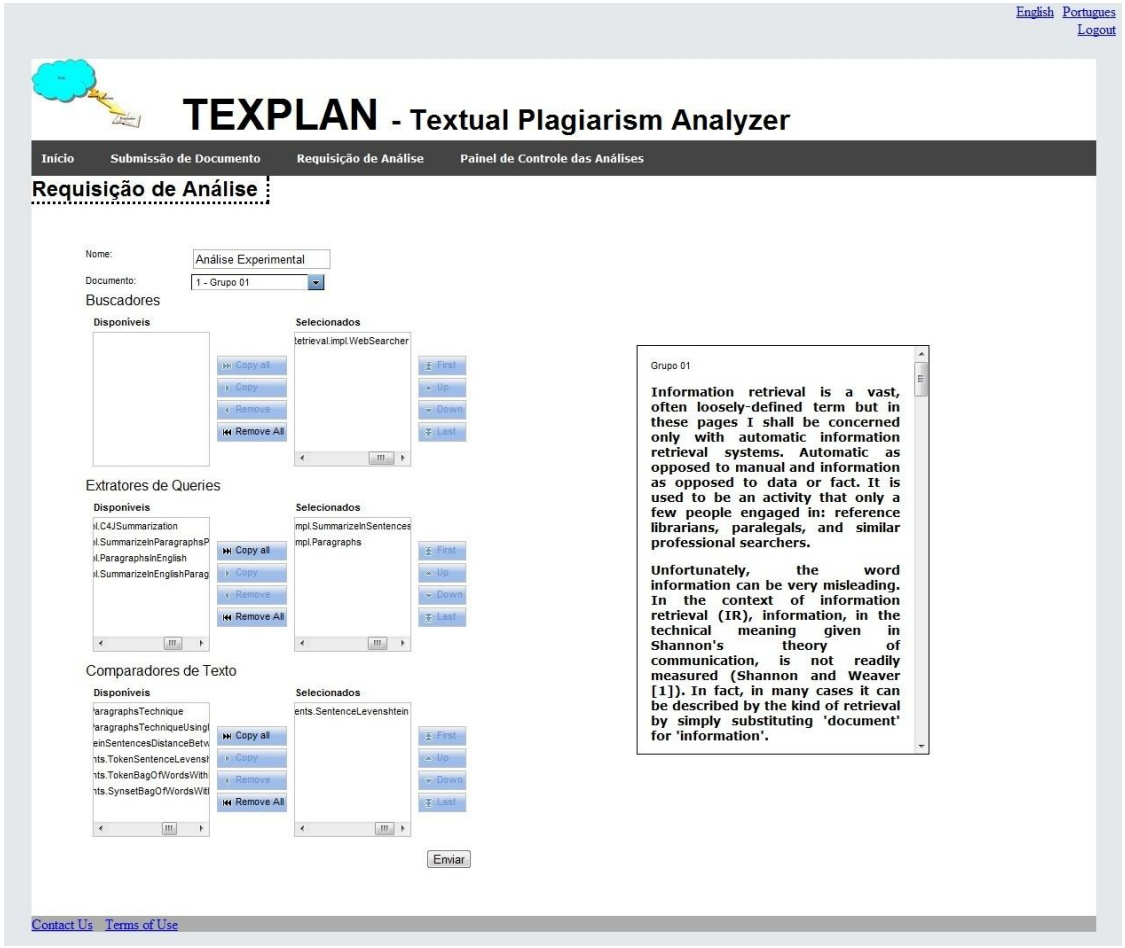


Figura 24 – Tela de requisição de análise.

No painel de controle das análises (Figura 25) é possível verificar o status das análises requisitadas, bem como ter um resultado, mesmo que parcial, das mesmas. Nessa tela é possível verificar quantas buscas (*query search*) e quantas comparações de documentos (*document analysis*) foram geradas, bem como o seu status. Deverá haver uma busca para cada par (consulta abstrata, ARD) e uma comparação de documentos para cada par (documento retornado, ACD).

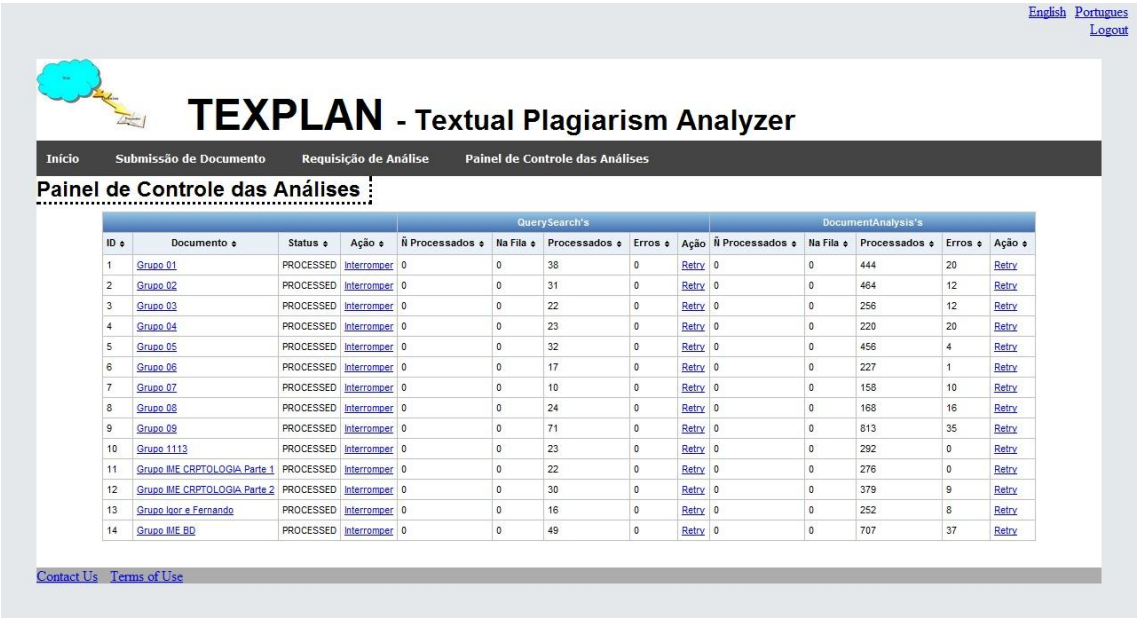


Figura 25 – Painel de controle de análises.

A qualquer momento é possível visualizar os resultados de uma análise. A tela de visualização de documento analisado (Figura 26) permite verificar que partes de um texto tem maior probabilidade de ser plágio a partir das similaridades até então detectadas. Para tal, o documento é apresentado como uma espécie de mapa de calor, e, quanto maior o índice de similaridade de um trecho do documento, mais intensa será a cor vermelha ao fundo do mesmo.

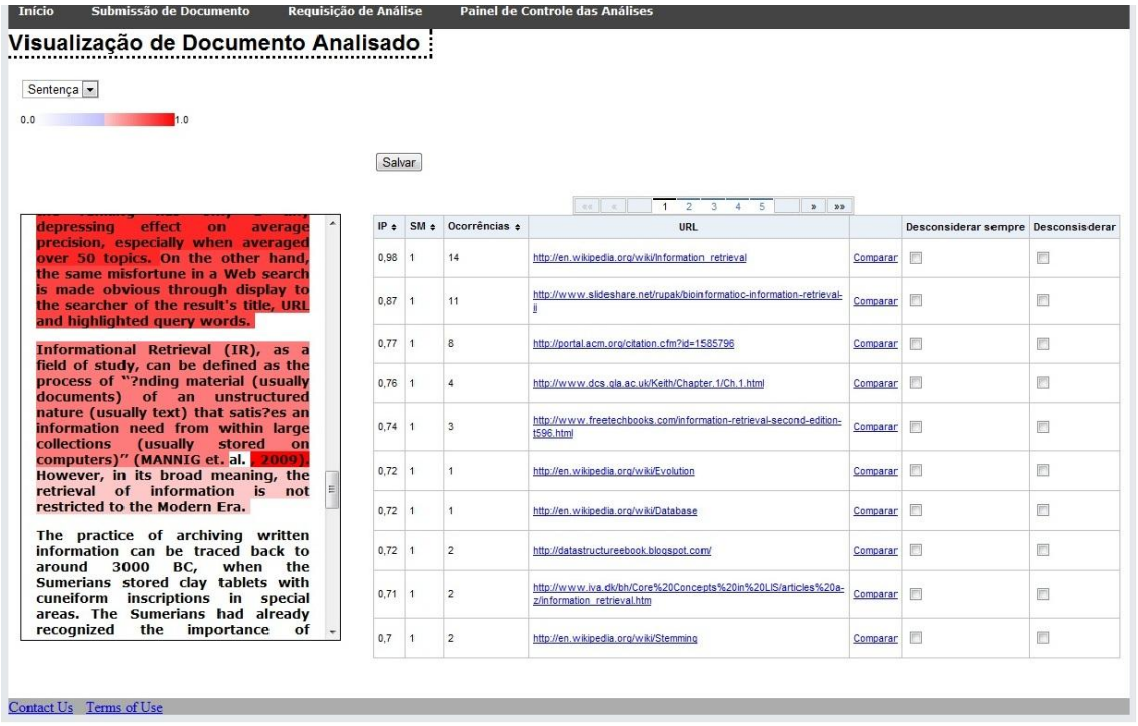


Figura 26 – Tela de visualização de documento analisado.

A tela de visualização de documento analisado também exibe uma listagem com os documentos similares encontrados durante a análise, sendo inicialmente ordenados pelo índice de plágio encontrado a partir da comparação do documento em questão com o documento em análise, além do valor máximo de similaridade encontrado e do número de vezes que o documento foi retornado pelas execuções das consultas. A tela também permite marcar os documentos que deverão ser desconsiderados nos resultados daquela análise, e também os que deverão ser desconsiderados em análises futuras daquele usuário.

A partir da lista de documentos similares encontrados, é possível acessar a tela de comparação de documentos (Figura 27). Na qual o analista de plágio pode visualizar as similaridades encontradas entre dois documentos, bem como fazer uma avaliação de cada similaridade. Cada similaridade pode ser classificada pelo analista como plágio, não plágio e não sei informar. Dessa forma, as similaridades marcadas como plágio por um ou mais analistas poderão ser utilizadas como fundamento para afirmações mais precisas.

Figura 27 – Tela de comparação/similaridades de documentos.

5.2.6 Limites da implementação

Alguns pontos da arquitetura, considerados não essenciais, não foram implementados durante o trabalho dessa dissertação devido ao tempo restrito. Os pontos em questão serão desenvolvidos em trabalhos futuros, e estão descritos a segui.

5.2.6.1 Filtro de consultas

O filtro de consultas tem papel relevante na melhoria do desempenho de uma análise, uma vez que elimina consultas desnecessárias ou com baixa probabilidade de acerto. No entanto, sua não implementação, não invalida o trabalho principal da arquitetura como comparador de configurações de algoritmos e detector de plágio.

5.2.6.2 Reconfigurador de Comparações

A função principal do reconfigurador de comparações é tratar casos em que parágrafos tenham sido divididos durante o processo de plágio e casos de má interpretação dos documentos digitais, como PDF`s, pela ferramenta. Entretanto, a validação da

arquitetura focou na detecção de plágio em nível de sentença, o que permitiu deixar a implementação desta etapa da arquitetura para os trabalhos futuros.

5.2.6.3 Gerador de Relatórios

Nesta etapa a ferramenta auxilia a geração de relatórios que comprovem um plágio. Essa etapa é importante no auxílio ao Analista de Plágio, que deverá analisar melhor os resultados e oficializados, porém não impossibilita a validação da arquitetura e realização dos experimentos.

5.3 Experimento

5.3.1 Objetivo

O objetivo do experimento é demonstrar que a arquitetura proposta é capaz de detectar plágio e de comparar o desempenho de diferentes algoritmos para esse fim.

5.3.2 Descrição

Uma turma de pós-graduação *Strictu Sensu* foi submetida a um trabalho no qual fora dividida em grupos de duas a três pessoas. Cada grupo deveria ler o capítulo 2 (Plágio) e então produzir um documento, sobre o tema *Information Retrieval*, com as seguintes características:

- O documento deveria conter 5 (cinco) parágrafos por membro do grupo;
- A cada cinco parágrafos, um deveria ser original e os demais plágio;
- O documento deveria ser escrito em inglês, bem como as fontes plagiadas;
- As fontes do plágio deveriam ser encontradas a partir da ferramenta de busca Google;
- O documento deveria possuir no mínimo três fontes de plágio distintas.

Cada grupo também deveria registrar, em um documento a parte, as fontes utilizadas, bem como as técnicas de plágio aplicadas em cada parágrafo.

Os documentos produzidos foram submetidos ao protótipo da ferramenta de detecção de plágio proposta nesta dissertação, em diferentes configurações de algoritmos, e os resultados encontrados foram comparados entre si e com os resultados da submissão dos mesmos documentos a outras ferramentas de detecção de plágio disponíveis na Internet.

5.3.3 Execução e resultados

A turma de pós-graduação produziu 15 documentos, dos quais 13 foram submetidos à ferramenta. Dois dos documentos foram descartados por não respeitarem as regras básicas pré-estabelecidas, contendo plágio com tradução da língua portuguesa para a inglesa e parágrafos plagiados de documentos únicos com a mesma sentença repetida na maioria dos parágrafos.

Todos os documentos produzidos pela turma possuem um cabeçalho com data, nome da instituição de ensino, nomes dos alunos participantes entre outras informações dispensáveis a análise. Conforme a seção 4.1.1 dessa dissertação, as consultas com informações do cabeçalho poderiam ser filtradas automaticamente, porém não foi possível implementar essa etapa da arquitetura a tempo da execução dos experimentos, tendo a mesma ficou como trabalho futuro (vide seção 5.2.6.1). Devido a essa não implementação, os cabeçalhos foram retirados manualmente, antes da submissão dos documentos.

Na base de dados da ferramenta foram relacionadas, aos documentos submetidos, as URL's utilizadas pelos alunos na construção do plágio. Dessa forma, foi possível construir consultas SQL que retornassem os indicadores necessários. No entanto, houve trabalhos que utilizaram em alguns de seus parágrafos plágio através de tradução e plágio de fonte não disponível na internet. Esses parágrafos foram tratados como originais durante o calculo dos indicadores do experimento.

O experimento se propõe a testar, principalmente, as etapas de extração de consultas, análises de documentos e geração de relatórios da arquitetura proposta. Não é intuito deste experimento, ou mesmo desta dissertação, mostrar que o protótipo TEXPLAN possui melhores resultados que os das ferramentas disponíveis no mercado, mas sim demonstrar que a arquitetura funciona e que suas fases são complementares umas as outras. O experimento também se propõe a demonstrar que a arquitetura atende sua proposta de ser uma base para o estudo e teste de algoritmos que possam beneficiar o processo de detecção de plágio.

A execução do experimento foi dividida nas etapas: extração de consultas e recuperação de documentos, análise de documentos e geração de relatórios. Por fim, é feita uma avaliação dos resultados obtidos.

5.3.3.1 Extração de Consultas e Recuperação de Documentos

Todos os documentos foram submetidos à primeira e à segunda fases da arquitetura, de modo a comparar os resultados da execução de três AEC's. O primeiro AEC gera como consultas um texto simples para cada parágrafo existente no documento analisado. Cada consulta será o conteúdo do parágrafo correspondente.

No segundo AEC, cada consulta gerada é um texto simples representando uma das sentenças pertencente ao grupo das 30% mais relevantes do texto analisado. As sentenças mais relevantes são determinadas através de uzma implementação simplificada do algoritmo de sumarização de Luhn (vide seção 5.2.3.1). Enquanto no terceiro AEC todas as sentenças são utilizadas como consultas.

Na fase de recuperação de documentos, as consultas são executadas na ferramenta de busca Google através de um ARD.

Foram feitas 39 análises (três para cada documento). Para cada análise *a*, foram calculados os indicadores *Precision* e *Recall*, conforme as fórmulas abaixo.

$$Precision(a) = \frac{|retornados(a) \cap plagiados(d)|}{retornados(a)}$$

$$Recall(a) = \frac{|retornados(a) \cap plagiados(d)|}{plagiados(d)}$$

, onde $retornados(a)$ é o conjunto de documentos retornados pelo ARD e $plagiados(d)$ é o conjunto de documentos plagiados pelo documento *d*.

A Figura 28 apresenta o *recall* obtido em uma das 39 análises, enquanto a Figura 29 apresenta o *precision* de cada análise. Observando as figuras, é possível verificar que os algoritmos utilizados neste experimento tem um bom *recall*, no entanto baixíssimo *precision*. Embora não seja o intuito deste trabalho encontrar os melhores algoritmos para implementação da arquitetura, cabe ressaltar que já era esperado um indicador de precisão baixo. O indicador de precisão tende a melhorar conforme uma análise avança nas fases da arquitetura.

Esta fase do experimento tem o intuito de demonstrar que é possível utilizar a ferramenta como laboratório para testes de diferentes algoritmos. Um bom AEC deve ter um alto *recall*, aumentando assim as chances de detecção de plágio ao final do

processo. No entanto, embora o *precision* melhore conforme a análise avança pela arquitetura, um valor de precisão baixo, afeta o desempenho da ferramenta, desperdiçando tempo computacional. Dessa forma, devem-se avançar as pesquisas por algoritmos que melhorem o *precision* perdendo o mínimo ou nada de *recall*.

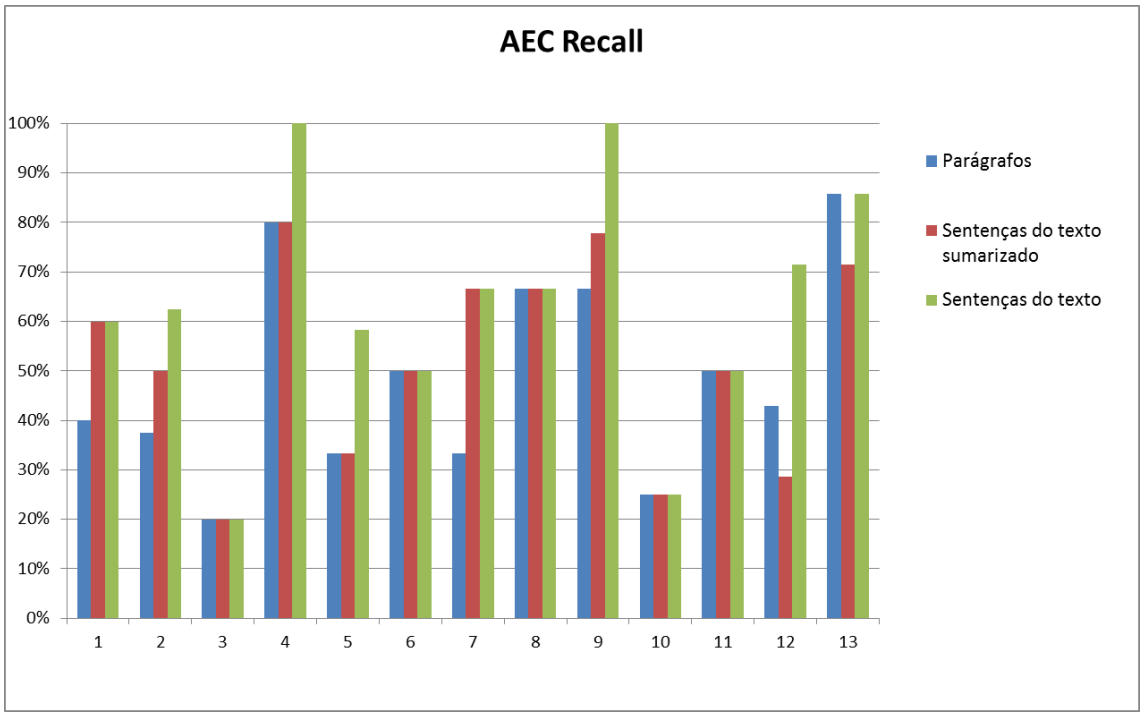


Figura 28 – Recall de cada análise por AEC.

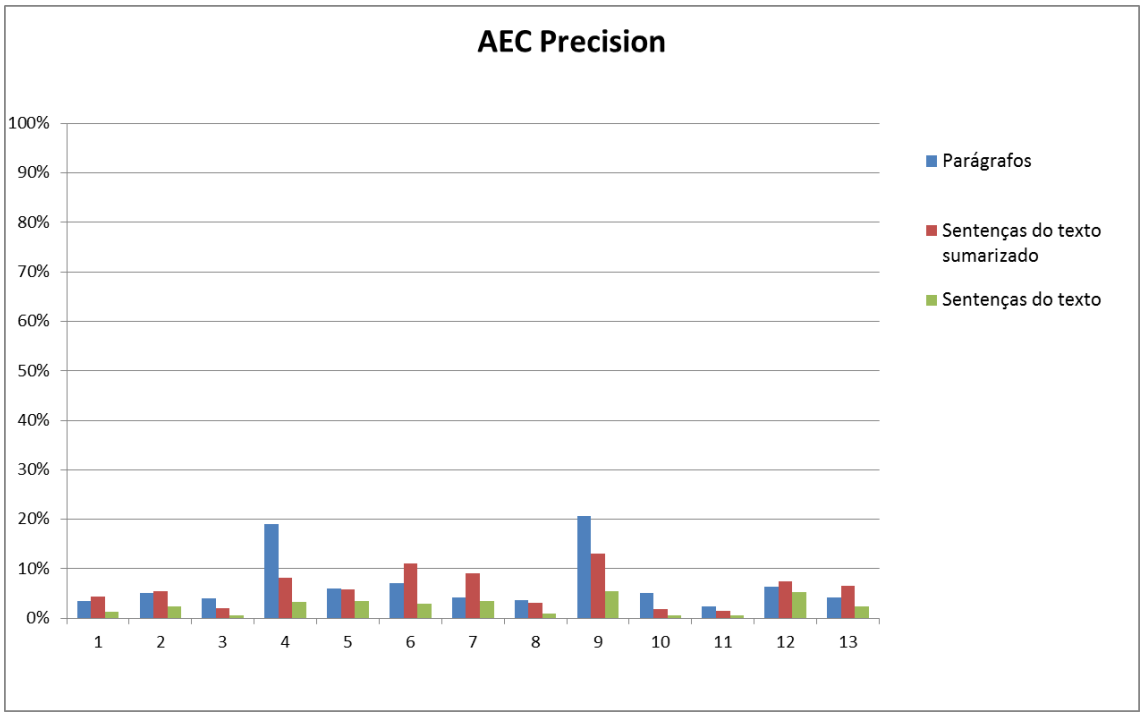


Figura 29 – Precision de cada análise por AEC.

A Tabela 12 apresenta o *precision* e o *recall* médios encontrados em cada tipo de análise executada.

Tabela 12 – Resultados dos Algoritmos Extratores de Consultas.

AEC	*Precision* Médio	*Recall* Médio
1- Parágrafos	7,00 %	48,54 %
2- Sentenças do texto sumarizado	6,08 %	52,26 %
3- Sentenças do texto	2,50 %	62,79 %

As conclusões apresentadas na subseção a seguir validam o ambiente fornecido pela ferramenta como propício a comparações de algoritmos.

5.3.3.1.1 Comparação de algoritmos

O terceiro AEC apresenta o melhor desempenho em recuperar os documentos originais. No entanto, sua precisão é bem inferior a dos demais AEC's, sendo aproximadamente 64% inferior a do primeiro, e aproximadamente 59% inferior a do segundo. Uma precisão baixa indica um alto percentual de documentos analisados sem necessidade, o que acarreta em maior desperdício de tempo computacional.

Por outro lado, o primeiro e o segundo AEC's conseguiram recuperar, respectivamente, 77,3% e 83,23% do número de documentos originais recuperados pelo terceiro. O que demonstra que é possível criar algoritmos que mantenham valores de revocação (*recall*) próximos, porém com precisão superior.

A Tabela 13 apresenta comparações entre os AEC`s utilizados. Na tabela é possível verificar que o uso de algoritmos mais sofisticados como os algoritmos de sumarização podem trazer muitos benefícios a precisão, com perda proporcionalmente inferior para o *recall*. A tabela também demonstra que a arquitetura está apta a ser utilizada para validação e comparação de algoritmos extratores de consultas para recuperação de documentos plagiados.

Tabela 13 – Comparação entre AEC's.

	Melhoria *Precision* Médio	**Perda *Recall* Médio**
AEC 1 x AEC 3	180,00%	-22,69%
AEC 2 x AEC 3	143,20%	-16,77%
AEC 1 x AEC 2	15,13%	-7,12%

A Figura 28 mostra que algumas análises obtiveram 100% de *recall*, o que demonstra uma boa perspectiva para esses algoritmos. No entanto, faz-se necessárias maiores análises para identificar os reais motivos de documentos plagiados não estarem sendo recuperados nas demais análises. Uma das suspeitas seria a de que o número máximo de documentos retornados, por Consulta Abstrata, estaria baixo, visto que, este experimento só considerou os primeiros cinco resultados do Google, enquanto só na primeira página de resultados do mesmo estão disponíveis dez resultados, sendo esses grandes candidatos a fontes de plagiado.

5.3.3.2 Análise de Documentos

A partir dos resultados das fases anteriores foram executados quatro ACD's. O primeiro utilizou uma variação do algoritmo de Levenshtein (vide seção 5.2.3.3.1), onde foi calculada a distância entre sentenças, ao invés de palavras, e comparadas palavras, ao invés de letras.

O segundo ACD também implementou Levenshtein calculando a distância entre sentenças, porém, antes, retirando as *stopwords* das sentenças e transformando as palavras em seus radicais através do algoritmo de Porter (vide seção 5.2.3.3.2).

O terceiro ACD utilizou *bag of words* para comparar sentenças por *token*. Neste algoritmo foram eliminadas as stopwords, e as palavras restantes foram transformadas em seus radicais a partir do algoritmo de Porter (vide seção 5.2.3.3.3).

O último ACD, bem como o anterior, utilizou *bag of words* para comparação de sentenças por *synset*. Para tal, as *stopwords* foram eliminadas e cada palavra foi substituída pelo seu respectivo *synset*, obtido através da WordNet (vide seção 5.2.3.3.4).

Todos os ACD's utilizaram MySQL Fulltext (GOLUBCHIK, 2003) para recuperar as cinco sentenças, do documento comparado, mais próximas de cada sentença do documento suspeito. Dessa forma, não foi necessário comparar cada sentença do documento suspeito com todas as sentenças do outro documento.

Apenas similaridades com valores acima de 0.5 foram consideradas.

Os treze documentos suspeitos foram submetidos a quatro análises cada, distinguindo-se uma da outra pelo ACD utilizado. Em cada análise foi utilizado o terceiro AEC da etapa anterior do experimento, que utiliza todas as sentenças como consultas, bem como o ARD que executa buscas no Google.

Nesta etapa do experimento, o *Precision* e o *Recall* foram calculados conforme as formulas a seguir.

$$Precision(a) = \frac{|duplas\_trechos\_similares(a) \cap duplas\_trechos\_plagio(d)|}{duplas\_trechos\_similares(a)}$$

$$Recall(a) = \frac{|duplas\_trechos\_similares(a) \cap duplas\_trechos\_plagio(d)|}{duplas\_trechos\_plagio(d)}$$

, onde $duplas\_trechos\_similares(a)$ é o conjunto de duplas de trechos similares encontrados durante a análise *a* e $duplas\_trechos\_plagio(d)$ é o conjunto de duplas de trechos formados por um trecho do documento suspeito *d* e por um trecho, de outro documento, que tenha lhe dado origem. Entretanto, devido a limitações dos dados obtidos sobre a massa de documentos do experimento, só se tem conhecimento do(s) documento(s) origem de um determinado parágrafo. Sendo assim, para este experimento, foram consideradas como interseções, entre os conjuntos das fórmulas, as similaridades que correlacionem trechos de parágrafo pré-determinado nos dados da massa como plagiado, com qualquer trecho do documento sinalizado na mesma massa.

A Tabela 14 apresenta o *precision* e o *recall* médios encontrados na execução de cada análise sobre os treze documentos suspeitos.

Tabela 14 - Resultados da comparação entre Algoritmos Comparadores de Documentos.

ACD	*Precision* Médio	*Recall* Médio
1- Levenshtein de sentenças	15,72 %	59,16 %
2- Levenshtein de sentenças por *token* sem *stopwords*	14,36 %	60,40 %
3- *Bag of words* por *token* sem *stopwords*	10,85 %	62,90 %
4- *Bag of words* por *synset* sem *stopwords*	11,04 %	60,71 %

No experimento executado, os quatro algoritmos obtiverem resultados semelhantes quando considerado o *recall*, com um pequeno destaque para o terceiro ACD (*bag of words* por *token*), que em três dos treze documentos teve *recall* superior aos demais e obteve valores iguais aos demais nos outros dez, como mostra o gráfico na Figura 30. No entanto, os algoritmos que utilizaram Levenshtein, o primeiro e o segundo ACD, notoriamente obtiveram melhor precisão quando comparados aos algoritmos que utilizam *bag of words,* como mostra a média na Tabela 14 e o *precision* por documento no gráfico da Figura 31.

Uma melhor precisão dos ACD's culmina em menor número de validações a serem feitas pelo analista de plágio. No entanto, vale lembrar que essa precisão leva em consideração apenas os documentos originalmente utilizados como fontes de plágio. Deve-se lembrar de que a ferramenta pode detectar outras fontes, das quais os documentos, até então, considerados originais podem ter sido referenciados, plagiados ou até mesmo cometido plágio. Dessa forma, fica como trabalho futuro um experimento que considere o julgamento de analistas de plágio sobre as similaridades encontradas, e recalcule dos indicadores *precision* e *recall*.

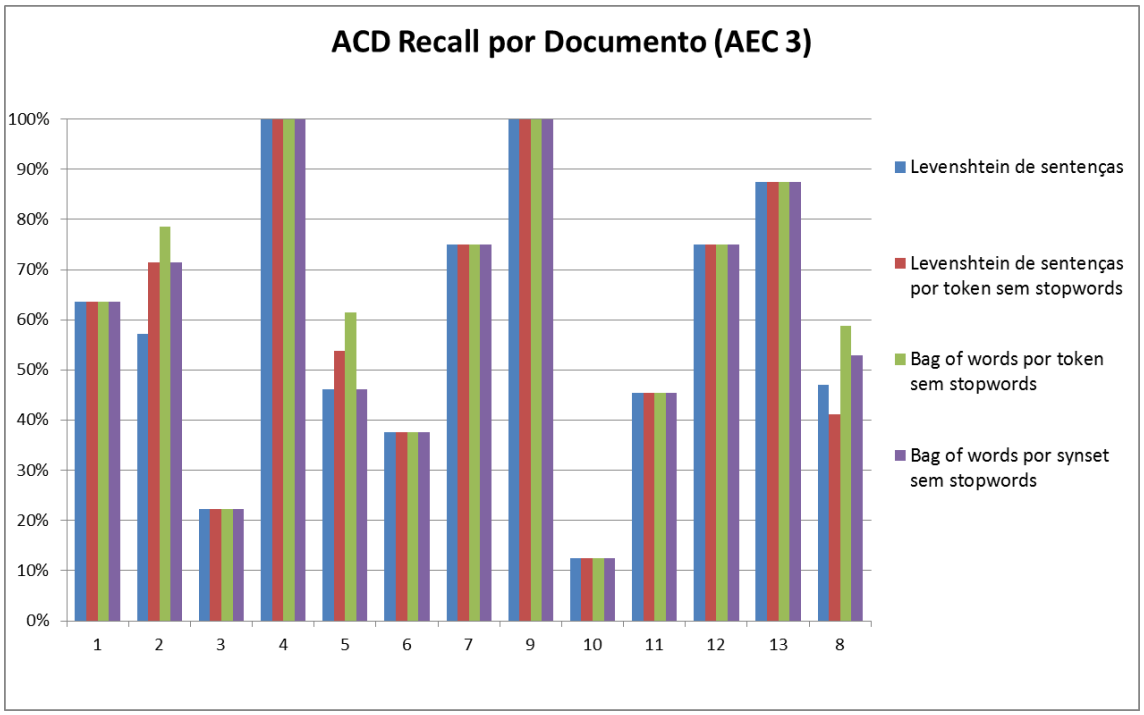


Figura 30 – *Recall* de cada ACD dividido por documento.

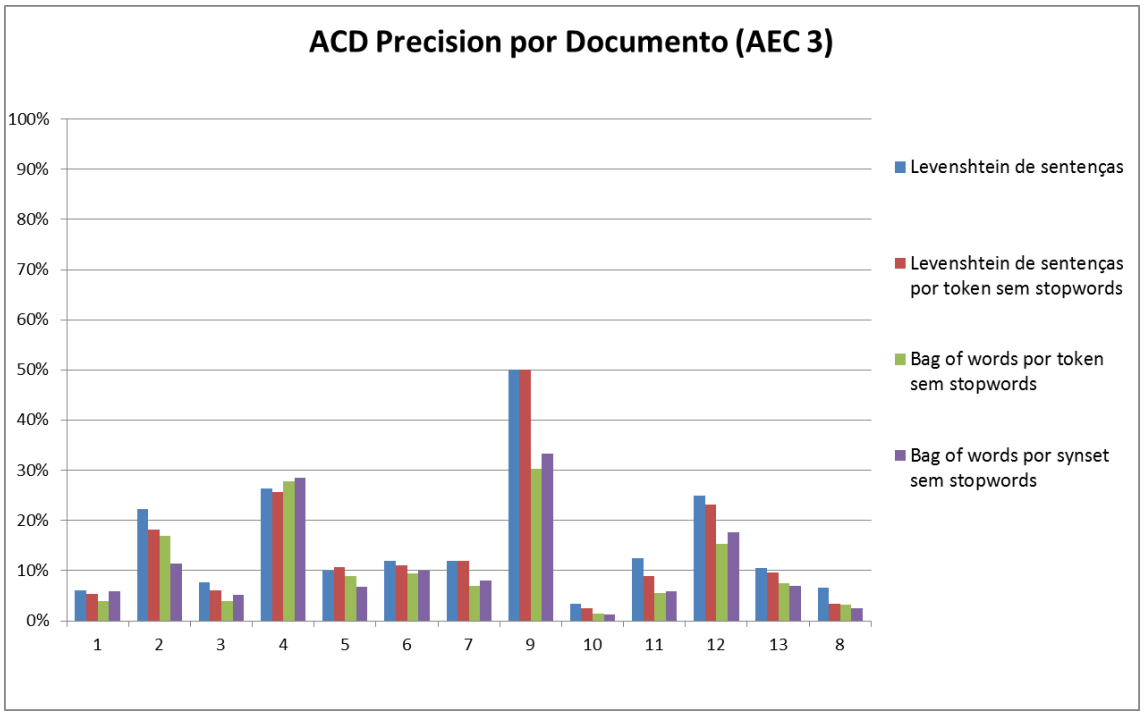


Figura 31 – *Precision* de cada ACD dividido por documento

5.3.3.3 Geração de Relatórios

A terceira fase deste experimento equivale à quinta fase da arquitetura. A geração de relatórios é a etapa final da análise. Nessa etapa são gerados os relatórios que serão

apresentados aos usuários. Os relatórios enfatizam similaridades e filtram o que há de mais importante a ser analisado pelo usuário, reduzindo assim o seu trabalho.

Nesta etapa do experimento foram gerados relatórios obtidos através eliminação de similaridades que não pertencessem a um grupo seleto de documentos fonte de plágio. Esses grupos foram criados a partir de um ranking de documentos mais similares ao documento suspeito. Esse ranking foi ordenado a começar pelos documentos que obtivessem o maior índice de plágio, calculado através da fórmula apresentada na seção 5.2.4.

5.3.3.3.1 Relatório com os dez documentos mais similares

O primeiro relatório considerou os dez documentos mais similares ao documento analisado. A partir desse grupo de documentos foram calculadas as mesmas fórmulas de *precision* e *recall* da fase anterior do experimento. Como resultado, obtivemos o demonstrado na Figura 32 e na Figura 33.

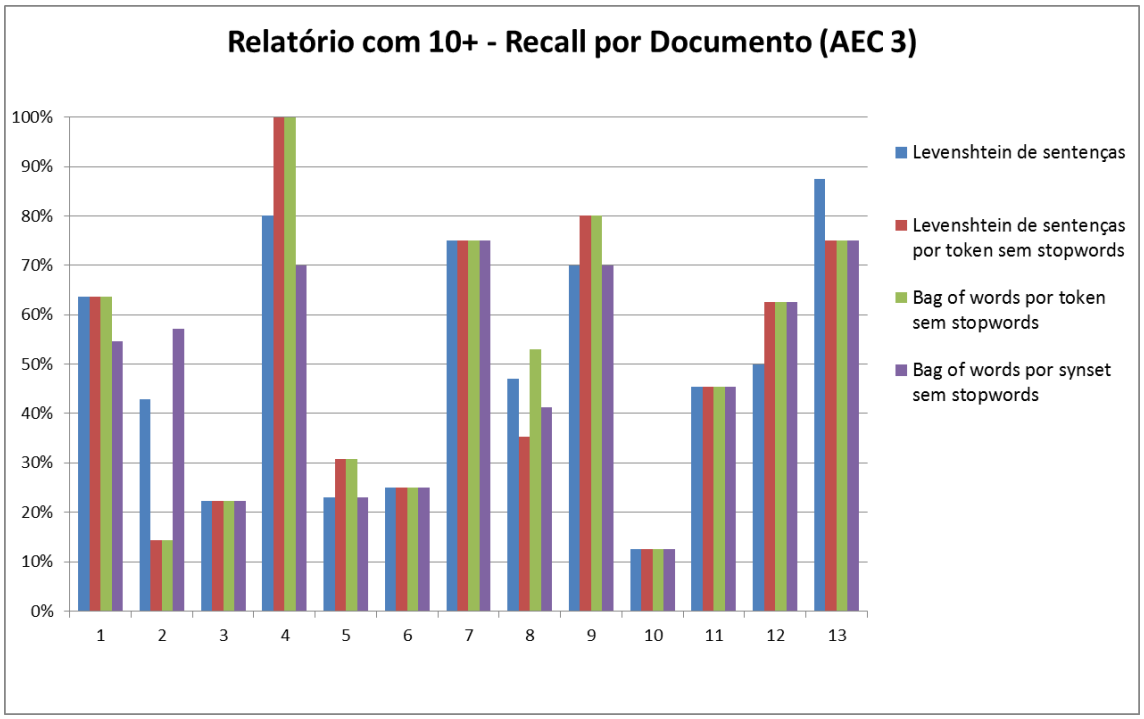


Figura 32 – *Recall* por análise dos relatórios com os 10 documentos mais similares.

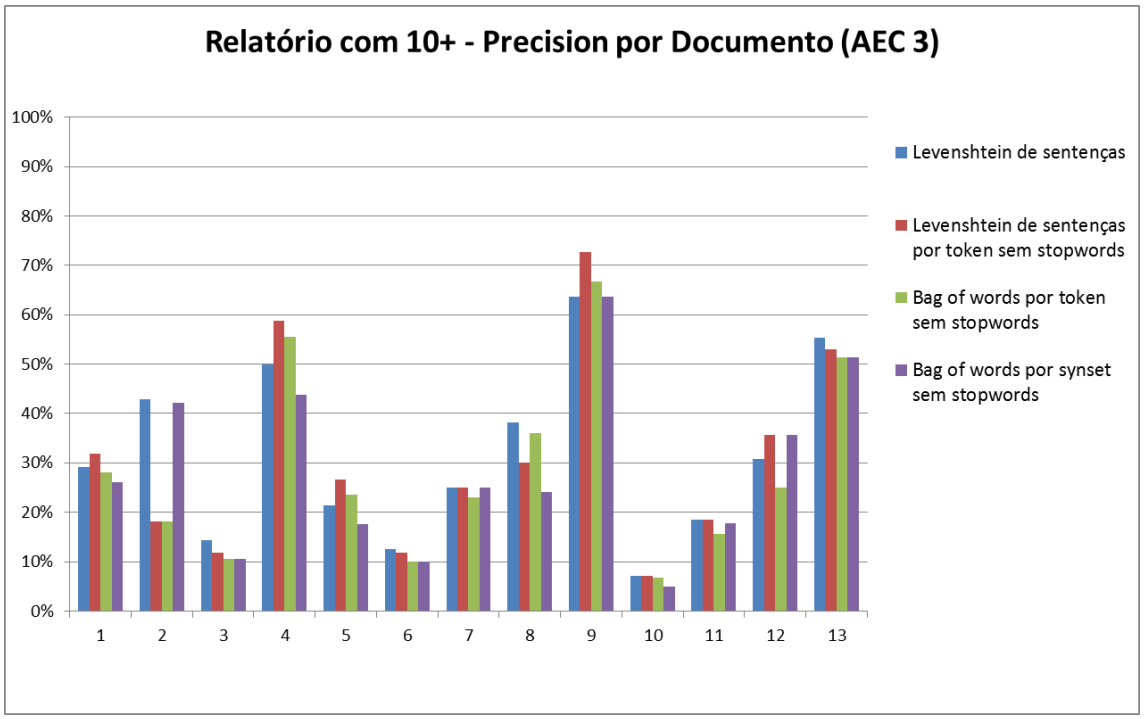


Figura 33 – *Precision* por análise dos relatórios com os 10 documentos mais similares.

5.3.3.3.2 Relatório com os cinco documentos mais similares

O mesmo foi feito para o grupo dos cinco documentos mais similares a cada documento suspeito segundo o ACD utilizado. A Figura 34 e a Figura 35 apresentam o *recall* e o *precision* encontrados em cada análise, respectivamente.

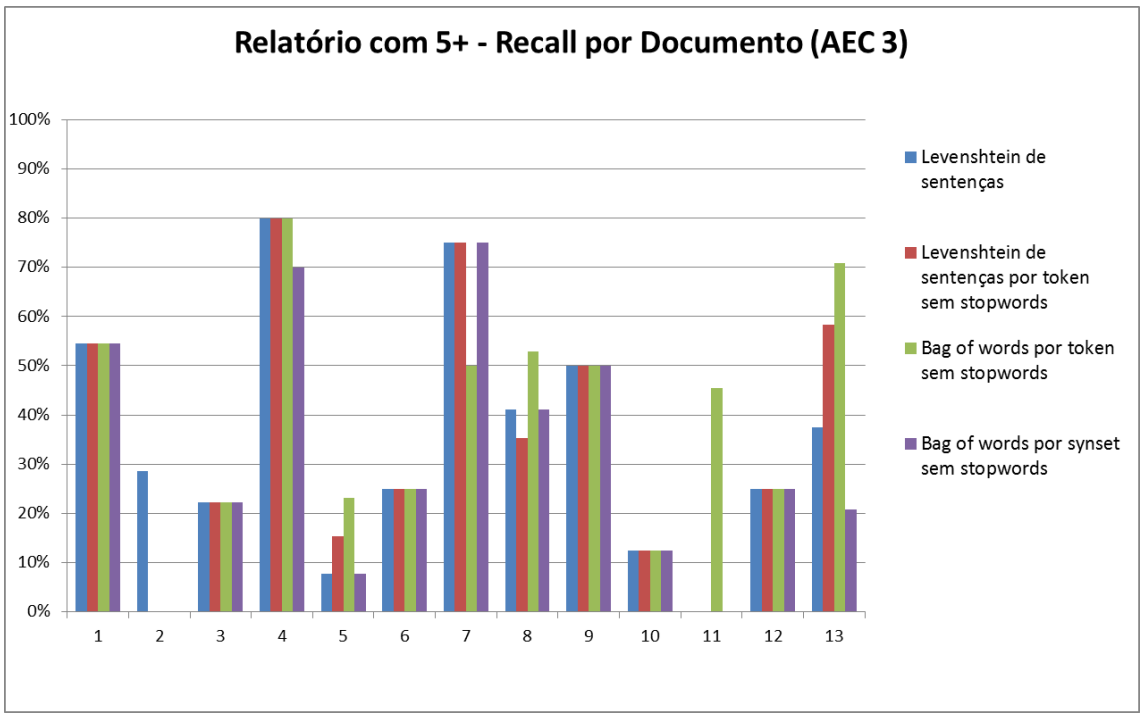


Figura 34 – *Recall* por análise dos relatórios com os 5 documentos mais similares.

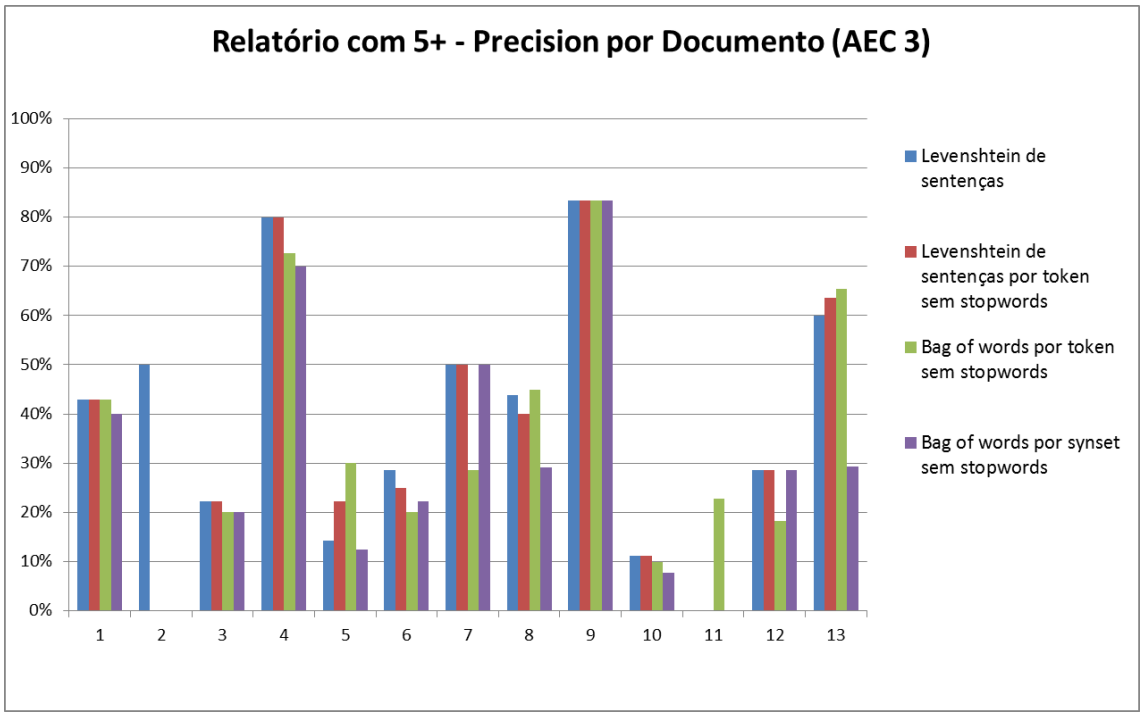


Figura 35 – *Precision* por análise dos relatórios com os 5 documentos mais similares.

5.3.3.3.3 Relatório com os dois documentos mais similares

Por fim, o último relatório considera apenas os dois documentos mais similares ao documento suspeito. O *recall* e o *precision* podem ser encontrados na Figura 36 e na Figura 37.

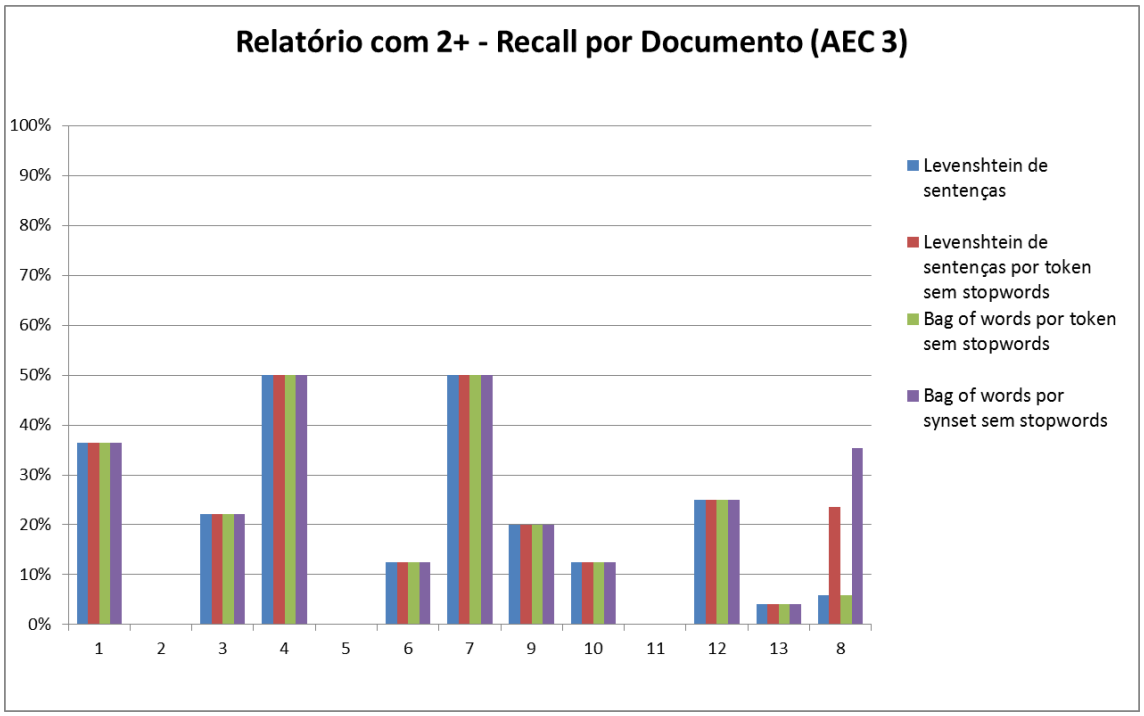


Figura 36 – Recall por análise dos relatórios com os 2 documentos mais similares.

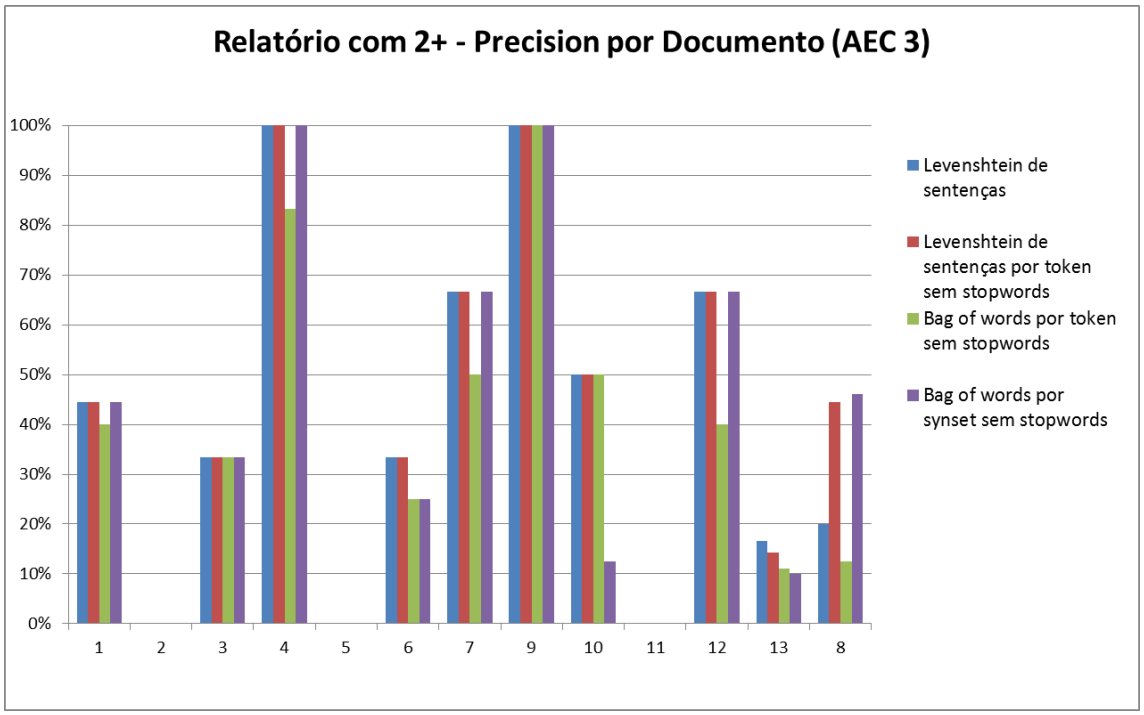


Figura 37 – Recall por análise dos relatórios com os 2 documentos mais similares.

5.3.3.4 Avaliação de resultados

Neste experimento, uma fase da análise cuja participação humana é essencial, a fase Análise de Plágio, não foi contemplada. Devido ao tempo restrito, não foi possível para este experimento utilizar análise humana.

No entanto, foi possível verificar o crescimento da precisão com que a arquitetura detecta plágios conforme uma análise avança através dela, como mostra a Figura 38. Em três dos ACD's utilizados o *precision* médio pode ultrapassar 50% ao final da análise.

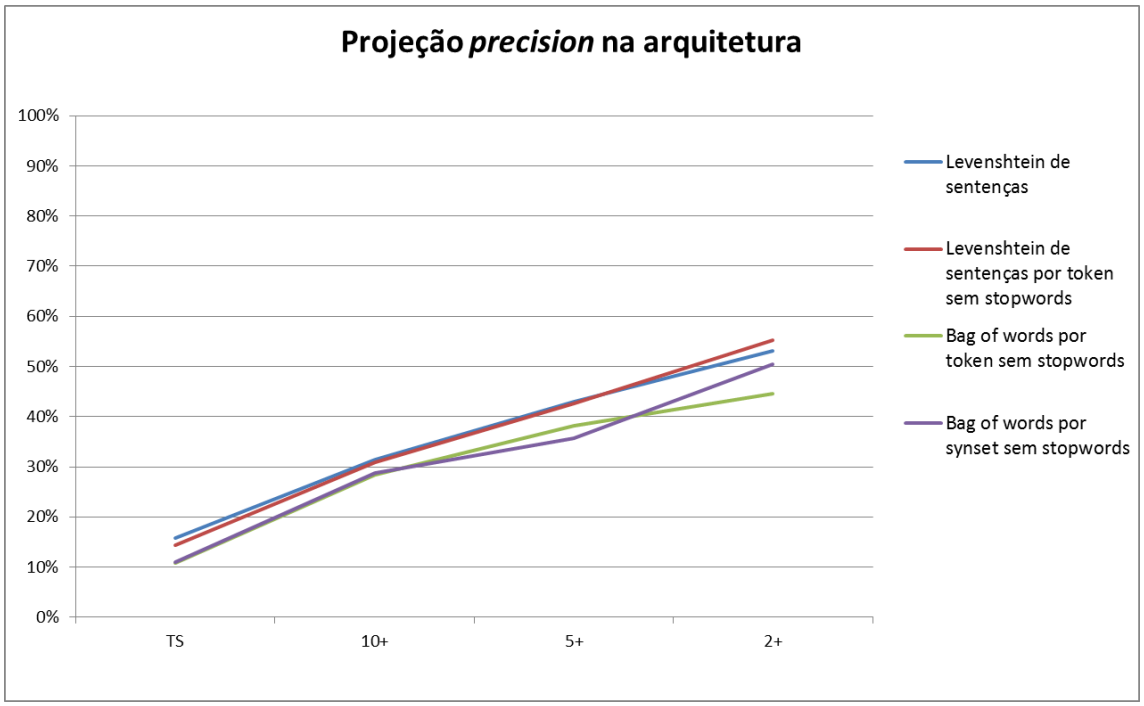


Figura 38 – Evolução do *precision* no experimento.

Cabe ressaltar que o intuito deste experimento e deste protótipo não é obter resultados melhores do que os das ferramentas já existentes, mas sim demonstrar o funcionamento da arquitetura proposta, que será usada em futuras pesquisas acadêmicas.

Também se observou, durante os experimentos, que muitos dos documentos cadastrados, como fontes oficiais do plágio, também estão disponíveis em outros endereços da Internet, ou tem seu conteúdo referenciado, e até mesmo plagiado, em outros endereços. O que também contribui para os baixos valores de *precision*. Sendo assim, na seção de trabalhos futuros, é proposto um novo experimento, no qual analistas de plágio julguem as similaridades encontradas para que seja possível obter indicadores mais adequados de *precision* e *recall*.

6 Conclusões e trabalhos futuros

Esta dissertação apresentou uma proposta de arquitetura para um sistema de detecção de plágio multi-algoritmo e foi capaz de validá-la através da implementação de um protótipo, batizado de Textual Plagiarism Analyzer, ou, simplesmente, TEXPLAN, e da execução de um experimento.

O experimento utilizou os esforços de uma turma da cadeira de Busca e Recuperação da Informação, da pós-graduação s*trictu sensu* do Programa de Engenharia de Sistemas e Computação (PESC) da COPPE/UFRJ. Esses esforços possibilitaram o uso de plágios reais, porém com fontes conhecidas, para que indicadores *precision* e *recall* pudessem ser determinados em cada fase de análise.

Foi possível verificar durante a validação da arquitetura que a implementação da mesma é factível e capaz de atender a proposta inicial de servir como base para testes de novos algoritmos. O experimento também demonstrou que as etapas da análise são complementares, filtrando os documentos e similaridades encontradas a cada etapa, e, dessa forma, melhorando a precisão das similaridades apresentadas ao usuário do sistema, como mostrou a Figura 38.

O experimento evidenciou resultados interessantes e, até mesmo, aceitáveis para uma versão inicial de um sistema de detecção de plágio. Além disso, o sistema foi utilizado na análise de uma monografia de conclusão de curso de graduação, a qual foi qualificada como plágio, e o autor foi notificado pelo seu orientador antes que a mesma fosse publicada.

Também foi possível verificar a eficácia de diferentes algoritmos como Levenshtein e *bag of words* na detecção de plágio.

6.1 Trabalhos futuros

Não houve tempo hábil para que nesta dissertação de mestrado toda a arquitetura fosse implementada no protótipo TEXPLAN, conforme destacado na seção 5.2.6. A correção das atuais limitações, já listadas, deverá ser feita como parte de um trabalho futuro.

O tema desta dissertação abre muitas possibilidades de pesquisa. Dentre eles podemos destacar alguns como a investigação de algoritmos para extração de consultas (AEC's),

que melhorem a precisão dos documentos retornados pela ferramenta, mantendo um valor de *recall* semelhante ou superior aos já obtidos. Quanto maior a precisão, menor o número de downloads e comparações necessários à execução de uma análise.

Entre as similaridades muitas vezes são detectados trechos de textos referenciados. Dessa forma, é interessante a pesquisa por técnicas que identifiquem referências corretas e as filtrem durante o cálculo dos índices de plágio e na exibição para o usuário. Surgindo assim, mais uma proposta de trabalho.

Dentre os trabalhos futuros também pode ser lembrado experimento que validará de forma mais precisa os resultados da ferramenta. Esse experimento consiste em submeter as similaridades entre textos, encontradas pela ferramenta, à análise humana, e avaliar a precisão da ferramenta, de seus índices de plágio e filtros de consulta.

A comparação com outras ferramentas também é um ponto interessante a ser tocado, no entanto, não houve tempo hábil para submeter os dados manualmente nas ferramentas existentes.

7 Referências Bibliográficas

BARRÓN-CEDEÑO, A.; ROSSO, P. **On automatic plagiarism detection based on n-grams comparison**. Advances in Information Retrieval. [S.l.]: Springer Berlin / Heidelberg. 2009. p. 696-700.

BENEDÍ, J. M.; BARRÓN-CEDEÑO, A.; ROSSO, P. **Reducing the Plagiarism Detection Search Space on the Basis of the Kullback-Leibler Distance**. 10th International Conference on Computational Linguistics and Intelligent Text Processing. [S.l.]: [s.n.]. 2009. p. 523-534.

CLOUGH, P. **Plagiarism in natural and programming languages: An overview of current tools and technologies**. University of Sheffield. [S.l.]. 2000.

DAS, D.; MARTINS, A. F. T. **A Survey on Automatic Text Summarization**. Literature Survey for the Language and Statistics II course at CMU. [S.l.]: [s.n.]. 2007.

DUARTE, F. R. **Identificação de reuso em documentos digitais.** 2011. 119f. Dissertação (Mestrado em Engenharia de Sistemas e Computação) - COPPE, UFRJ, Rio de Janeiro. 2011.

ENCYCLOPÆDIA BRITANNICA INC. **The New Encyclopedia Britannica**. 15. ed. [S.l.]: Encyclopædia Britannica, Inc, v. Micropædia, VIII, 1984. 20 p.

GIBALDI, J. **MLA Handbook for Writers of Research Papers**. 6ª. ed. New York: Modern Language Association of America, 2003.

GOLUBCHIK, S. **MySQL Fulltext Search**. International PHP Conference. Frankfurt, Alemanha: [s.n.]. 2003.

HOWARD, R. M. A Plagiarism Pentimento. **Journal of Teaching Writing**, v. v11 nº 2, p. 233-245, 1992.

IPARADIGMS. Plagiarism prevention technology. **Plagiarism.org**, 2011a. Disponivel em: <http://plagiarism.org/plag_solutions.html>. Acesso em: 15 Maio 2011.

IPARADIGMS. Plagiarism Detection and Content Verification for Admissions. **Turnitin for Admissions**, 2011b. Disponivel em: <http://www.turnitinadmissions.com/>. Acesso em: 16 Maio 2011.

KUHN, H. W. The Hungarian Method for the assignment problem. **Naval Research Logistics Quarterly**, v. 2, p. 83-97, 1955.

LANCASTER, T.; CULWIN, F. **A visual argument for plagiarism detection using word pairs**. Plagiarism: Prevention, Practive and Policy Conference. [S.l.]: [s.n.]. 2004.

LANCASTER, T.; CULWIN, F. Classification of plagiarism detection engines. **ITALICS**, 4 Issue 2, maio 2005.

LEUNG, C.-H.; CHAN, Y.-Y. **A natural language processing approach to automatic plagiarism detection**. Conference On Information Technology Education. Nova York: Association for Computing Machinery. 2007. p. 213-218.

LEVENSHTEIN, V. Binary codes capable of correcting deletions, insertions, and reversals. **Soviet Physics Doklady**, v. 10, p. 707-710, 1966.

LUHN, H. P. The Automatic Creation of Literature Abstracts. **IBM Journal of Research Development**, 2, 1958. 159-165.

LUKASHENKO, R.; GRAUDINA, V.; GRUNDSPENKIS, J. **Computer-based plagiarism detection methods and tools:** an overview. Proceedings of the 2007 international conference on Computer systems and technologies. [S.l.]: ACM. 2007. p. 40:1-6.

MANBER, U. **Finding Similar Files in a Large File System**. USENIX Technical Conference. São Francisco, California: [s.n.]. 1994. p. 1-10.

MARTINS, C. B. et al. **Introdução à sumarização automática**. UFSCAr. [S.l.], p. 38. 2001.

MAURER, H.; KRAPPE, F.; ZAKA, B. Plagiarism - A Survey. **Journal of Universal Computer Science**, v12, nº 8, 2006. 1050-1084.

MILLER, G. A. **WordNet:** A Lexical Database for English. Communications of the ACM. [S.l.]: [s.n.]. 1995. p. 39-41.

NOYNAERT, J. E. **Plagiarism Detection Software**. Midwest Instruction and Computing Symposium. [S.l.]: [s.n.]. 2006.

ORACLE CORPORATION. Manual de Referência do MySQL. **MySQL**, 2011a. Disponivel em: <http://dev.mysql.com/doc/refman/4.1/pt/fulltext-search.html>. Acesso em: 25 Agosto 2011.

ORACLE CORPORATION. MySQL Internal Algorithms. **MySQL Forge**, 2011b. Disponivel em: <http://forge.mysql.com/wiki/MySQL_Internals_Algorithms>. Acesso em: 25 Agosto 2011.

ORACLE CORPORATION. Java Message Service API Overview. **Oracle.com**, 2011c. Disponivel em: <http://www.oracle.com/technetwork/java/overview-137943.html>. Acesso em: 28 Maio 2011.

PALEO, B. W. **An Approximate Gazetteer for GATE based on Levenshtein Distance**. 12TH ESSLLI STUDENT SESSION PROCEEDINGS. Dublin, Irlanda: [s.n.]. 2007. p. 197-208.

SILVA, A. K. L. D.; DOMINGUES, M. J. C. D. S. Plágio no meio acadêmico: de que forma alunos de pós-graduação compreendem o tema. **Perspectivas Contemporâneas**, v. 3, Nº 2, 2008. ISSN 1980-0193.

STEIN, B.; LIPKA, N.; PRETTENHOFER, P. Intrinsic plagiarism analysis. **Language Resources and Evaluation**, p. 1-20, 2010.

SUN MICROSYSTEMS, INC. Basic JMS API Concepts. **Oracle.com**, 2002. Disponivel em: <http://download.oracle.com/javaee/1.3/jms/tutorial/1_3_1-fcs/doc/basics.html>. Acesso em: 28 Maio 2011.

THE TRUSTEES OF PRINCETON UNIVERSITY. WordNet Statistics. **WordNet A lexical database for English**, 2011. Disponivel em: <http://wordnet.princeton.edu/wordnet/man/wnstats.7WN.html>. Acesso em: 25 Agosto 2011.

TRIPOLITIS, K. **Automatic Detection of Plagiarism**. University of Sheffield. [S.l.]. 2002.

VARELLA, A. N. **COOPRACTICE - Comunidades de prática virtuais apoiadas por ontologias**. 2007. 123f. Dissertação (Mestrado em Engenharia de Sistemas e Computação) - COPPE, UFRJ, Rio de Janeiro. 2007.

WEBER-WULFF, D. **Test cases for plagiarism detection software**. 4th Plagiarism Conference. [S.l.]: [s.n.]. 2010.