Identificador de plágio para o Adessowiki

Lucas Ikeda França
Orientador: Prof. Roberto Hirata Jr.

Motivação

Plágio é uma questão que preocupa o meio acadêmico. Em ambientes de educação a distância, principalmente aqueles baseados na tecnologia *wiki*, o problema torna-se ainda maior pois não há controle sobre a forma como os trabalhos são produzidos. Desta forma, procura-se evitar ao máximo sua ocorrência e, posteriormente, faz-se necessária sua identificação. Quando o número de trabalhos é grande, ferramentas automáticas facilitam essa tarefa.
O Adessowiki [1] é uma ferramenta de educação a distância desenvolvida em Python, com o *framework* Django, onde é possível administrar (acessar, editar, organizar) uma disciplina e seu conteúdo de forma colaborativa. Foi desenvolvido em uma parceria entre o CTI (Centro de Tecnologia da Informação Renato Archer) e a Faculdade de Engenharia Elétrica e de Computação da Unicamp. Ele vem sendo utilizado por diversas instituições de ensino em disciplinas de introdução à programação, visão e processamento de imagens, computação gráfica e processamento paralelo em *GPUs*. Até o início deste trabalho, não havia uma ferramenta de identificação de plágio associada ao Adessowiki.

Objetivo

A proposta deste trabalho é desenvolver uma ferramenta que facilite a detecção de plágio no Adessowiki. Conteúdo do tipo código-fonte, ou texto em linguagem natural devem ser tratados independentemente nessa identificação.
A saída para o usuário é um relatório onde se destacam os casos de possível plágio, assim como os trechos comuns que leventaram suspeitas.

Adessowiki

Implementação

Dentre as diversas soluções possíveis para resolver o problema, foi implementada a técnica de Winnowing [2]. A ideia central é gerar *fingerprints* para cada texto, e compará-los segundo algum critério (como uma função de *hash*) a fim de identificar os trechos onde possa haver ocorrido plágio.
Cada tipo de conteúdo é tratado separadamente. Para textos em linguagem natural, foi feita uma implementação local do Winnowing. O exemplo abaixo é uma aplicação desta técnica a um texto em linguagem natural:

texto A: 'Et omnia dicta fortiora si dicta latine.'

reduzido: 'etomniadictafortiorasidictalatine'

k-gramas:
'etomniadictafor', 'tomniadictafort', 'omniadictaforti', 'mniadictafortio', 'niadictafortior', 'iadictafortiora', 'adictafortioras', 'dictafortiorasi', 'ictafortiorasid', 'ctafortiorasidi', 'tafortiorasidic', 'afortiorasidict', 'fortiorasidicta', 'ortiorasidictal', 'rtiorasidictala', 'tiorasidictalat', 'orasidictalati', 'orasidictalatin', 'rasidictalatine'

hashes:
33, 88, 9, 23, 36, 92, 82, 46, 36, 68,15, 13, 91, 46, 75, 86, 62, 11, 20

fingerprints:
9 , 23, 36, 13 ,46

texto B: 'Omnia dicta fortiora, si graecis conscript.'

reduzido: 'omniadictafortiorasigraecisconscript'

k-gramas:
omniadictaforti', 'mniadictafortio', 'niadictafortior', 'iadictafortiora', 'adictafortioras','dictafortiorasi', 'ictafortiorasig', 'ctafortiorasigr', 'tafortiorasigra', 'afortiorasigrae', 'fortiorasigraec', 'ortiorasigraeci', 'rtiorasigraecis', 'tiorasigraecisc', 'iorasigraecisco', 'orasigraeciscon', 'rasigraeciscons', 'asigraecisconsc', 'sigraecisconscr', 'igraecisconscri','graecisconscrip', 'raecisconscript'

hashes:
9, 23, 36, 92, 82, 46, 39, 92, 71, 59, 34, 18, 60, 41, 6, 30, 68, 44, 56, 35, 72, 71

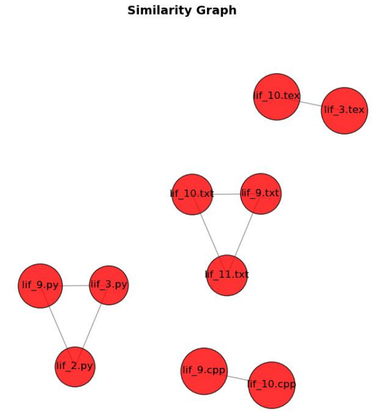
fingerprints:
9, 23, 36, 39, 6, 30

valores comuns: 9, 23, 36 → **trecho comum detectado:** omnia dicta fortiora

Como estratégia simplificadora, conteúdos do tipo código-fonte são processados pelo Moss [3], uma ferramenta online para detecção de plágio que também utiliza o Winnowing para comparar as entradas. Seu pré-processamento é mais sofisticado e suporta diversas linguagens de programação, entre elas Python e C++.
Foi implementado um *parser* para tratar a entrada e a saída desse programa, de modo que os resultados pudessem ser integrados aos relatórios gerados.

IME - Instituto de Matemática e Estatística

A imagem abaixo mostra a saída do programa, que é publicada no Adessowiki:




lif_3.py and lif_9.py may be suspicious. Details: http://moss.stanford.edu/results/420021655/match0.html
lif_10.cpp and lif_9.cpp may be suspicious. Details: http://moss.stanford.edu/results/420021655/match1.html
lif_2.py and lif_9.py may be suspicious. Details: http://moss.stanford.edu/results/420021655/match2.html
lif_2.py and lif_3.py may be suspicious. Details: http://moss.stanford.edu/results/420021655/match3.html
lif_10.tex and lif_3.tex may be suspicious. Details: http://moss.stanford.edu/results/420021655/match4.html
lif_10.txt and lif_9.txt may be suspicious. They share the following substring:
ates that a monkey hitting keys at random on a typewriter keyboard for an infinite amount of time will almost surely type a given text, such as the complete works of William Shakespeare.

Resultados

A ferramenta produzida é multiplataforma e pode ser facilmente integrada ao Adessowiki.
Ela ainda está em fase de testes e não foi utilizada nos cursos. Porém, nos testes, as suspeitas de plágio foram corretamente detectadas, isto é, o *software* apontou-as no relatório e, numa análise posterior, constataram-se as suspeitas de plágio.

Agradecimentos

Prof. Roberto de Alencar Lotufo (DCA - FEE - Unicamp)
Prof. Rubens Campos Machado (CTI Renato Archer)

Referências

[1] http://www.adessowiki.org
[2] Aiken, Schleimer, Wilkerson. *Winnowing: Local Algorithms for Document Fingerprinting*
[3] http://moss.stanford.edu

python