Universidade de São Paulo
Instituto de Matemática e Estatística

Edson Kiyohiro Miyahara

Genealogia Acadêmica Lattes

São Paulo
2011

Edson Kiyohiro Miyahara

Genealogia Acadêmica Lattes

Monografia apresentada junto ao curso de Bacharelado em Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo, como requisito parcial à obtenção do titulo de Bacharel.

Orientador: Prof. Dr. Roberto Marcondes Cesar Junior

Coorientador: Dr. Jesús Pascual Mena Chalco

São Paulo

2011

Agradecimentos

Agradeço aos meus orientadores, Prof. Dr. Roberto Marcondes Cesar Junior e Dr. Jesús Pascual Mena Chalco, pela oportunidade de trabalhar neste projeto e por todas as sugestões dadas ao trabalho. Agradeço à minha família por todo o apoio dado aos meus estudos.

Resumo

A genealogia acadêmica é utilizado para organizar, através de uma árvore de genealogia, pesquisadores por meio de suas relações de orientação ou supervisão. Uma árvore de genealogia acadêmica indica, comumente, a linhagem de um pesquisador.

Nesse contexto, este trabalho descreve uma ferramenta para geração automática da árvore de genealogia acadêmica para pesquisadores e acadêmicos cadastrados na Plataforma Lattes.

O trabalho está sendo desenvolvido em Python no ambiente Gnu/Linux. A ferramenta obtêm os Curriculos (CVs) da Plataforma Lattes a partir do ID Lattes fornecido e guarda-os em um *cache*, então processa e armazena-os em uma estrutura de dados que permite gerenciar e acessar rapidamente as informações dos CVs. Também será criado um algoritmo para a identificação de nomes similares evitando assim a ambiguidade entre nomes de uma pessoa escrita de formas diferentes. São gerados diferentes grafos em formatos que permitam a posterior visualização e análise por ferramenta externas/complementares.

Sumário

1 Introdução

Atualmente, com o grande crescimento de informações disponíveis na internet, torna-a um campo fértil para o desenvolvimento de diversas áreas da ciência da computação, dentre elas a mineração de dados (*data mining*) que vem ganhando grande destaque e importância nesse cenário.

Este trabalho foca sobre as informações contidas nos currículos Lattes, presente na Plataforma Lattes, para a geração automática da árvore de genealogia dos acadêmicos, sendo então possível realizar a sua visualização por meio de ferramentas complementares ou externas.

1.1 Motivação

Esta é uma proposta nova no Brasil e está inserido no contexto de prospecção de dados para análise de dados da genealogia acadêmica. Esta informação é de importante interesse para análise da inter-relação de orientações.

1.2 Objetivos

Neste projeto procuramos desenvolver uma ferramenta para geração automática da árvore de genealogia acadêmica para cientista e acadêmicos cadastrados na Plataforma Lattes e no intuito do desenvolvimento desse projeto, tem-se como principais objetivos:

- O desenvolvimento de um sistema de armazenamento dinâmico dos Currículos (CVs), com o intuito de reduzir o número de acessos á rede.
- Tratamento de similaridades entre nomes, o algoritmo tenta decidir se nomes escritos de formas diferentes pertencem á mesma pessoa, evitando assim possíveis ambiguidades com os nomes de pesquisadores.
- Desenvolver um algoritmo para geração da árvore de genealogia acadêmica descendente e ascendente.

- Exportação das informações para arquivos que permitam a visualização e análise por ferramentas externas e/ou complementares.
- Geração da árvore em um mapa geográfico utilizando a API do *Google Maps*.

2 Conceitos e Tecnologias

Essa seção pretende-se apresentar um visão geral sobre conceitos e algumas das diversas tecnologias utilizada no âmbito do desenvolvimento deste projeto.

2.1 Genealogia Acadêmica

A genealogia acadêmica tenta organizar cientistas e acadêmicos em uma árvore genealógica, segundo suas relações de orientação ou supervisão concluídas. Vale ressaltar que orientações e supervisões em andamento não são considerados. Uma descrição da ideia desde processo foi descrito brevemente em [2].

Existem alguns projetos nessa área como o projeto de genealogia matemática[1] da Sociedade Americana de Matemática que tem como objetivo compilar informações de todos os matemático do mundo, mas que também aceitam informações sobre pesquisadores de outras áreas [3]. As informações como nome completo, nome da universidade, ano de obtenção, titulo completo da dissertação e nome(s) do(s) orientador(s) e orientando(s) são exibidas em um página web e não há opção de visualização gráfica das relações de orientação. Veja na Figura 1 um exemplo de arvore de genealogia acadêmica de J. Bernoulli.

1 Disponível em http://genealogy.math.ndsu.nodak.edu

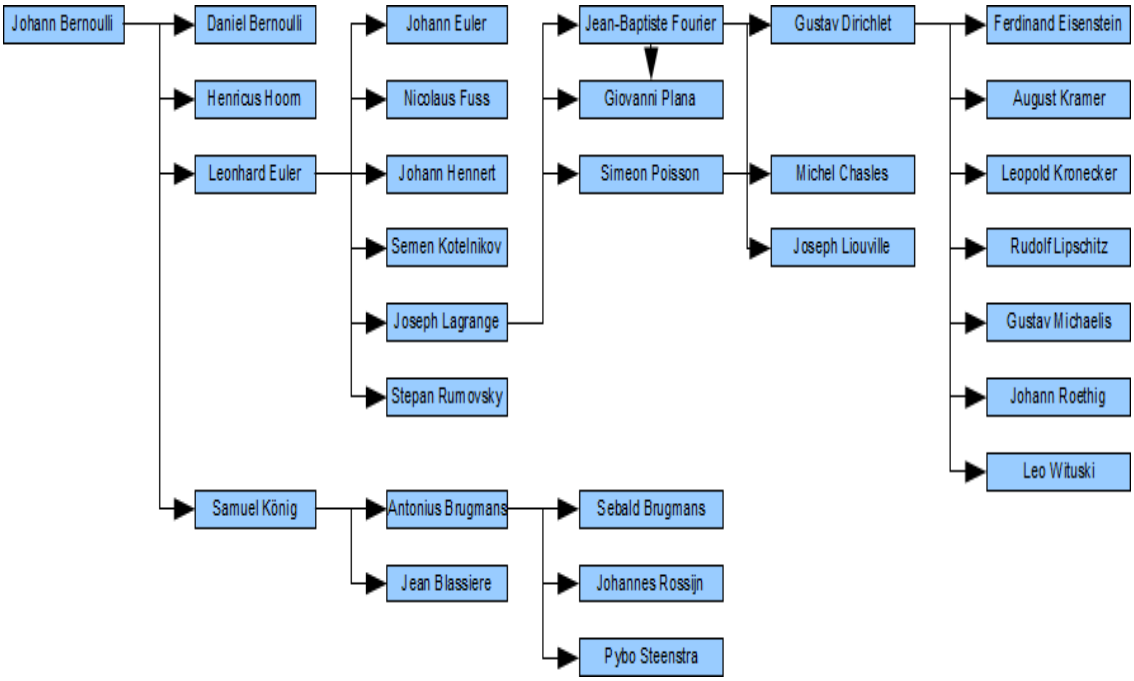

Figura 1: Exemplo de um árvore construída a partir de informações disponíveis em http://genealogy.math.ndsu.nodak.edu/id.php?id=53410

2.2 Plataforma Lattes[2]

A Plataforma Lattes é um sistema de informações mantido pelo Conselho Nacional de Desenvolvimento Científico e tecnológico (CNPq) que permite gerenciar informações curriculares de pesquisadores e acadêmicos da área de Ciência e Tecnologia de todo o Brasil.

Segundo o CNPq [4], esse sistema surgiu a partir da necessidade de se ter um sistema que possibilitasse a avaliação curricular de pesquisadores e gerar estatísticas sobre a distribuição da pesquisa no Brasil. No final de 1999, o CNPq lançou e padronizou o Currículo Lattes que é o currículo padrão da Plataforma Lattes, assim o Currículo Lattes tornou-se um padrão nacional apresentando uma grande riqueza de informações e uma crescente confiabilidade e abrangência.

Atualmente o Currículo Lattes encontra-se em uso em diversos países da América latina tais como Argentina, Colômbia, Equador, Peru entre outros [3].

2 Disponível em http://lattes.cnpq.br

2.3 *Web Mining* e *Data Mining*

Web mining (mineração na web) é a forma de utilização das técnicas de *data mining* (mineração de dados) para extrair informações úteis da web [6].

A *web mining* é dividido em três categorias:

- *Web content mining* (mineração de conteúdo da web) que tenta descobrir informações de interesse em documentos web, principalmente através da análise textual.
- *Web structure mining* (mineração da estrutura da web) que realiza a análise da interconectividade entre web sites.
- *Web usage mining* (mineração dos registros de navegação na web) tenta revelar padrões de acessos de transações web ou de registros de sessão do usuário gravados em arquivos de log.

Data mining ou prospecção de dados refere-se a extração de conhecimento informativo de um grande volume de dados, cujo principal objetivo é descobrir conhecimento escondido ou invisíveis, normalmente na forma de padrões [6].

2.4 ScriptLattes[3]

O ScriptLattes é uma ferramenta *open-sorce* sob licença GNU-GLP que está sendo desenvolvido no CCSL-IME/USP por Jesús P. Mena-Chalco e Roberto M. Cesar-Jr, escrita em Python, permite a geração automática de relatórios acadêmicos em formato HTML, considerando apenas informações cadastradas nos Currículos Lattes. O relatório é constituído de informações como produções bibliográficas, técnicas e artísticas, orientações, projeto de pesquisa, prêmios e títulos, grafos de colaborações e mapa de geolocalização [2, 5].

3 Disponível em http://scriptlattes.sourceforge.net

2.5 Graph-Tool[4]

Graph-Tool é um modulo Python, escrito por Tiago de Paula Peixoto sob licença GPL, que possibilita a manipulação e análise de grafos estáticos. O núcleo dos algoritmos e estruturas de dados são escritos em C++, utilizando Boost Graph Library (BGL) e meta-programação, as principais características são:

- Facilidade em criar e manipular grafos de forma arbitrária.
- Capacidade de associar informações arbitrarias à vértices, arestas e ao grafos em si através de um mapa de propriedades.
- Aplicação de filtros à vértices e/ou aresta de forma que eles aparentam ter sido removidos, mas podem ser facilmente recuperados.
- Facilidade em transformar grafos dirigidos em não dirigidos e vice-versa e de reverter a direção das aresta de grafos dirigidos.
- Capacidade de coletar vários tipos de estatísticas.
- Possui vários algoritmos topológicos.
- Capacidade de salvar arquivos em diversos formatos incluindo dot e GraphML.
- Geração de grafos randômicos.

2.6 Google Maps Api[5]

Google Maps API é um serviço gratuito [1], que permite a qualquer desenvolvedor incluir em seus *websites* e aplicações, informações e/o diagramas relacionados com localização geográfica. No nosso projeto utilizamos o API do *Google Maps* para, dado um Endereço Profissional, identificar as correspondentes coordenadas de latitude e longitude dos pesquisadores.

4 Disponível em http://projects.skewed.de/graph-tool
5 Disponível em http://code.google.com/apis/maps

3 Método de geração automática de árvores de genealogia

Na Figura 1 são mostrados os cinco processos considerados no sistema de geração de árvores de genealogia acadêmica.

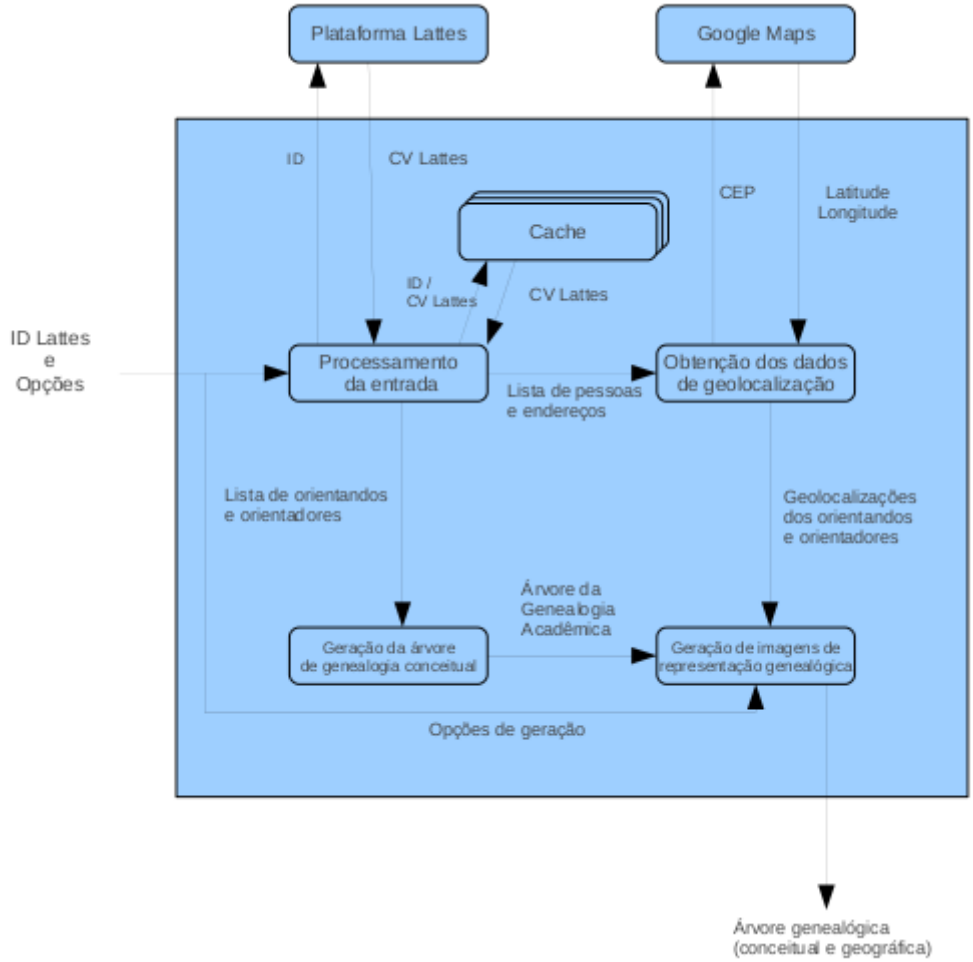

Figura 2: Fluxograma dos processos envolvidos.

3.1 Cache

É um módulo bastante simples, é responsável por armazenar os currículos que são obtidos da Plataforma Lattes para que, numa posterior consulta, estes currículos não precisem ser baixados novamente. Possui basicamente três métodos importantes, um método de verificação, um de armazenamento e outro de recuperação.

A presença do cache melhora significativamente o tempo de execução do programa, uma vez que os currículos presentes nele não precisam ser baixados novamente e que o tempo de acesso ao cache é menor que o tempo de obtenção dos currículos da Plataforma Lattes, já que está sujeito a atrasos de comunicação na rede.

Uma desvantagem da utilização do *cache* é que não há a atualização do currículos já presentes, ou seja, quando um currículo é atualizado na Plataforma Lattes e este já estava armazenado no cache, as mudanças não são refletidas no currículo armazenado. Para contornar esse problema, pode ser adicionado um procedimento para atualização dos Currículos Lattes, presentes no cache, com uma determinada frequência.

3.2 Processamento da entrada

Este módulo é responsável pelo processamento da entrada fornecida pela linha de comando, pelo pré-processamento dos currículos que é realizada pelo *parser* do scriptLattes e pela verificação de similaridade entre nomes.

A entrada da linha de comando é formada obrigatoriamente pelo ID Lattes e algumas opções que definem as características da árvore, tais como os tipos de orientações que serão exibidas na árvore, podendo ser pós-doutorado (PD) , doutorado (D) e mestrado (M) ou qualquer combinação deles, o padrão é a exibição de todos os tipos de orientação, e o número de níveis da árvore.

Os currículos obtidos tanto da Plataforma Lattes como do *cache* precisam ser pré-processados para podermos ter acesso às informações, esse pré-processamento é realizado pelo *parser* do scriptLattes. As informações de

relacionamento de orientação entre orientador e orientandos são mantidas em uma lista e junto com o currículo pré-processado são colocados em outra estrutura de dados e então inserida em uma lista que chamaremos de dicionário, dessa forma podemos evitar o reprocessamento dos currículos, por exemplo quando um indivíduo foi orientador por dois orientadores, então o currículo desse indivíduo seria pré-processado duas vezes, uma quando está-se verificando o orientador 1 e o outro quando verifica-se o orientador 2. Com o dicionário podemos checar se o currículo do indivíduo já foi pré-processado, antes de realizar esse procedimento.

O *parser* extrai as informações do currículo Lattes e armazena-os em uma estrutura de dados chamada *Membro*, a partir disso temos acesso a todas as informações que estavam contidas no currículo, mas neste trabalho estamos interessados apenas nas informações sobre o nome do pesquisador ou acadêmico, seu endereço profissional, nome e ID Lattes (se encontrado) dos seus orientandos, tipo de orientação e ano de conclusão e o ID Lattes (se encontrado) de seu(s) orientador(es). Essas informações sobre os orientandos e orientador(es) estão contidas em listas de orientações concluídas e um lista de IDs Lattes, respectivamente. Assim *Membro* é o currículo pré-processado.

Esse currículo pré-processado é inserido em uma estrutura de dados chamada *Indivíduo* que então é inserida no dicionário. A estrutura *Individuo* também possui um campo que guarda um lista de filhos, ou seja, orientandos. Cada elemento dessa lista, chamado *Dados*, possui um ID que indica a posição no dicionário onde podemos encontrar o seu currículo (se existir), nome, tipo de orientação, ano de conclusão e ID do pai. Esses elemento são inseridos nessa lista depois que o currículo de cada orientando da listas de orientação concluída for pré-processado.

Dessa forma, a partir da pesquisa sobre um pesquisador ou acadêmico, é construído o dicionário que então nos permite descobrir as relações de orientação. Veja na Figura 3 um esquema que simplifica o processo de criação do dicionário.

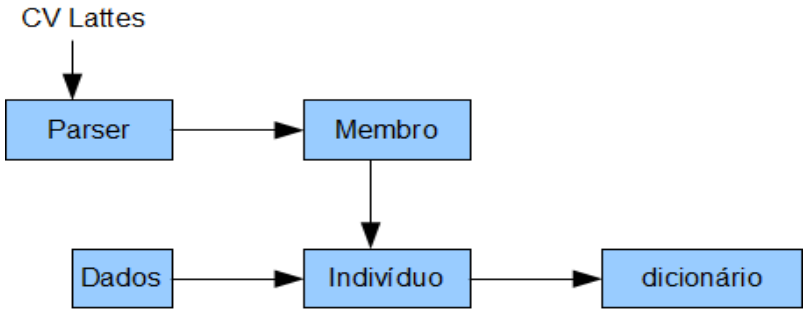

Figura 3: Esquema que exemplifica o processo de criação do dicionário.

Comumente, nos dados cadastrados na Plataforma Lattes, uma pessoa pode ter seu nome escrito de formas diferentes como por exemplo, ter abreviado seu primeiro nome ou seu nome do meio ou ambos ou até mesmo ocultado parte do nome, como mostra o exemplo abaixo:

Edson Kiyohiro Miyahara	*Edson K. Miyahara*
E. Kiyohiro Miyahara	*E. K. Miyahara*
Edson Miyahara	*E. Miyahara*

Diante dessas possibilidades como podemos identificar que esses nomes representam uma mesma pessoa?

O algoritmo abaixo pretende realizar essa tarefa. Ele parte da premissa de que todo nome fornecido tem o sobrenome como a última palavra do nome e é escrita somente com letras minúsculas ou maiúsculas sem acentos ou sinais gráficos. Assim inicia-se transformando os nomes recebidos para comparação, ou seja, nome1 e nome2 em array de palavras n1 e n2 tal que n1[0] é a primeira palavra de nome1 e n1[k-1], onde k é o número de palavra que forma nome1, é a última palavra, ou seja, o sobrenome. O mesmo vale para n2.

Após esse processo o algoritmo compara os sobrenomes, se forem iguais então comparamos o primeiro nome, se o resultado for negativo não significa, necessariamente, que se trata de pessoas diferentes, pois um dos nomes pode estar abreviado. Então verificamos se um dele é constituído de uma única letra, caso seja, checamos se são iguais. Só depois da confirmação de que o sobrenome e o primeiro nome são iguais, passamos a comparar os nomes do meio.

A comparação dos nomes do meio é realizada de forma um pouco diferente, comparamos apenas a primeira letra, uma vez que eles podem estar abreviados. Essa comparação é realizada pelo método auxiliar *ComparaAux* que recebe duas array de nomes, nome1 e nome2 que são, respectivamente, n1 e n2 obtidas anteriormente, iniciamos esse processo pelo nome que possui o menor número de palavras no nome do meio, pois uma vez que o nome e o sobrenome são iguais, estas palavras podem ter um correspondente no outro nome.

A ordem das correspondências corretas deve ser a mesma nos dois nomes, assim se nome1[i][0] é igual nome2[j][0] então a comparação de nome1[i+1][0] se inicia a partir de nome2[j+1][0] até nome2[m-2][0], onde m é o número de palavras em nome2. Lembra-se de que nome1[i][0] é a primeira letra da i-ésima palavra de nome1. Nesse processo de comparação contamos o número de correspondências corretas e calculamos a taxa de correspondência, considerarmos como nomes iguais se a taxa for maior ou igual a 90%, também consideramos como iguais se o número de palavras no nome do meio for igual a 0.

```
/* recebe 2 nomes completos */
ComparaNomes (nome1, nome2)
        n1 ← array de palavras de nome1
        n2 ← array de palavras de nome2
        se sobrenome de nome1 == sobrenome de nome2
                então se nome de nome1 == nome de nome2
                        então se ComparaAux (n1, n2)
                                retorna True
                senão se nome de nome1 ou nome2 está abreviado
                        então se a 1º letra de nome1 == à de nome2
                                então se ComparaAux(n1, n2)
                                        retorna True
        retorna False

/* recebe 2 arrays de nomes */
ComparaAux (nome1, nome2)
        se o nº de palavras em nome1 > à do nome2
                então nome1 ↔ nome2
        se o nº de palavras em nome1 - 2 == 0
                então retorna True
        k ← 1,
        corretos ← 0,
        n ← nº de palavras de nome1
        m ← nº de palavras de nome2
        para i em [1 … n-2] faça
                para j em [k ...m-2] faça
                        se nome1[i][0] == nome2[j][0]
                                então k ← j + 1, corretos ← corretos + 1, break
        se corretos/(n-2) >= 0.9
                então retorna True
                senão retorna False
```

O algoritmo *ComparaNomes* executa em tempo assintótico de O(nm), onde n e m representam o número de palavras que formam o primeiro nome e segundo nome, respectivamente. Veja na Figura 4 exemplos da execução do algoritmo.

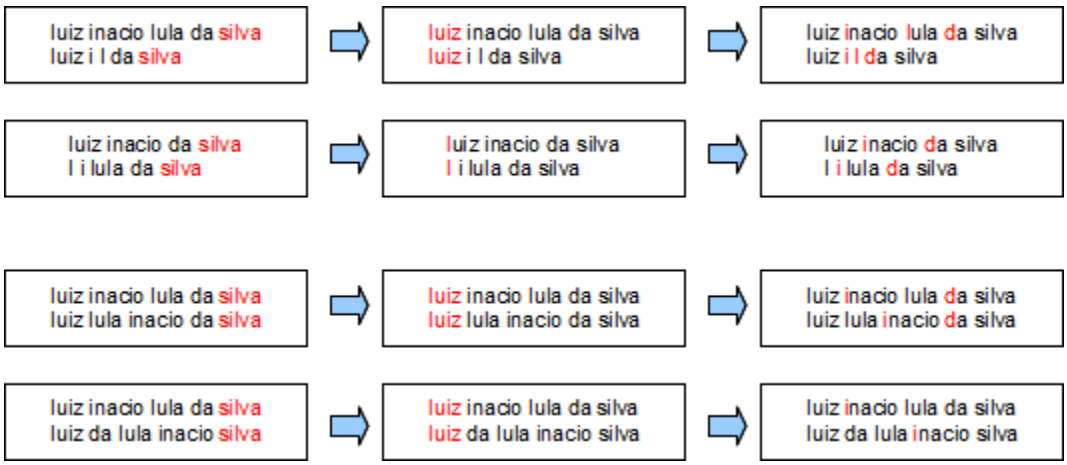

Figura 4: Exemplo da execução do algoritmo, os dois primeiros são exemplos em que os nomes são similares e os dois últimos não são.

3.3 Geração da árvore de genealogia conceitual

Na geração da árvore de genealogia conceitual é utilizada a ferramenta Graph-Tool que utiliza uma lista de adjacências como estrutura de dados para representar grafos.

Os métodos deste módulo utilizam as informações, coletadas pelo pré-processamento dos Currículos Lattes, que estão armazenadas em uma lista que guarda as relações de orientação entre orientador e orientando que encontra-se no dicionário com os currículos pré-processados. Neste módulo podemos gerar a árvore de genealogia descendente e ascendente.

Uma árvore de genealogia descendente é uma árvore que a partir do indivíduo pesquisado exibe seus orientandos e possivelmente os orientandos dos orientandos. Árvore de genealogia ascendente segue o sentido contrário da descendente, ou seja, a partir do indivíduo pesquisado exibe-se o orientador desse indivíduo e possivelmente o orientador do orientador.

O método para geração da árvore de descendência é realizado por um algoritmo de busca em profundidade, inicia-se percorrendo a lista, a partir do

indivíduo pesquisado e para cada orientando desse indivíduo é verificado se ele também é um orientador, ou seja, se tem orientandos. Se tiver orientandos, continua a busca até atingir o nível especificado pelo o usuário, para cada relação orientador-orientando é inserido uma aresta no grafo junto com o tipo de orientação e ano de conclusão como rótulo da aresta e o nome do orientando no vértice.

O processo para árvore de ascendência é muito semelhante, a partir do indivíduo pesquisado, insere-se uma aresta para cada relação de orientação com seus orientandos e também rótulo a aresta com o tipo de orientação e ano de conclusão e o vértice com o nome do orientando, e é realizado o mesmo processo para cada orientador do indivíduo pesquisado, e do orientador do orientador, assim sucessivamente até atingir o nível especificado pelo usuário. Veja na Figura 5 um exemplo genérico para a geração de uma árvore conceitual com três vértices usando o Graph-Tool.

```
tree = Graph()
v1 = tree.add_vertex()
v2 = tree.add_vertex()
v3 = tree.add_vertex()
tree.add_edge(v1,v2)
tree.add_edge(v1,v3)
```

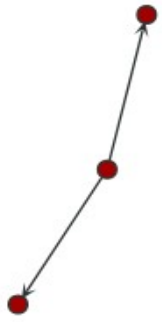

Figura 5: Exemplo genérico da geração da árvore conceitual utilizando o Graph-Tool.

Após termos o grafo, podemos aplicar filtros sobre ele para exibir apenas as informações que desejarmos, por exemplo se o usuário desejar ver uma árvore de genealogia acadêmica somente para mestrado. A aplicação do filtro inicia-se pela instanciação da classe *GraphView* e passando o grafo original como objeto a ser filtrado e o filtro que é uma função que irá atuar sobre os vértices e/ou arestas.

O grafo final é então salvo nos formato PNG e GraphML que é feita pela chamada ao método *graph_draw* do graph-tool, já para o formato JSON é

realizada pela chamada ao método *salvarGrafoEmJSON,* que recebe o grafo a ser salvo, o dicionário e o nome do arquivo. Ele percorre todos os vértices e para cada vértice, salva seu ID, que é obtido do grafo, e como vértices adjacente todos os IDs dos vértices que estão conectados a ele por uma aresta de saída, já que o grafo é orientado, também salva informações adicionais, como nome completo, tipo de orientação, ano de conclusão, coordenadas geográficas e endereço profissional.

3.4 Obtenção dos dados de geolocalização

A obtenção das coordenadas geográficas é realizada pelo método *obterCoordenadas* da classe Geolocalizador do scriptLattes, ele utiliza o endereço profissional obtido do currículo Lattes que é passado como argumento na requisição ao Google Maps, e retorna as coordenadas geográfica correspondentes: latitude e longitude. Esse método é chamado para cada para cada elemento do dicionário e as coordenadas são armazenadas na estrutura de dados desse elemento. Atualmente apenas consideramos o CEP, unidade federativa (UF) e nome do pais para identificar as coordenadas de latitude e longitude. Dada a falta de padronização nos CVs Lattes não consideramos como entrada o endereço profissional completo.

3.5 Geração de imagens de representação geográfica

A geração de imagens de representação geográfica é realizada por um script escrito em javascript, utilizando ajax para ter acesso aos dados gravados em um arquivo no formato JSON, entre esses dados estão as coordenadas geográficas.

Após a obtenção desses dados usamos a API do Google Maps , versão 3, para criarmos um mapa, realizada pela instanciação da classe *google.maps.Map,* então para cada vértice do grafo criamos uma marca através da instanciação da classe *google.maps.Marker* e os inserimos no mapa através do método *setMap.*

As criação das arestas são realizadas pela instanciação da classe *google.maps.Polyline* e inseridas no mapa pelo método *setMap.* A criação das aresta com seta é particularmente mais complicado, pois não há um método para criá-lo de forma automática, é preciso calcular a inclinação da polyline (aresta) e arredondá-lo para um múltiplo de 3, então carregar um icone correspondente através da instanciação da classe *google.maps.MarkerImage*, inseri-lo numa marca e depois no mapa.

As janela de informações são criadas no momento da criação das marca pela instanciação da classe *google.maps.InfoWindow* passando o conteúdo a ser exibido e associando uma janela a cada marca.

Após finalizada o todo processo, a imagem de representação geográfica pode ser visualizada através de um navegador web.

3.6 Visualização interativa

Para realizar a visualização interativa é utilizada uma ferramenta externa, o InfoVis Toolkit que é escrito em javascript. As informações sobre o grafo são obtidas do arquivo em formato JSON que são carregadas por ajax e então passada ao InfoVis Toolkit que carrega os dados JSON em sua estrutura, então basta estabelecer algumas opções como tipo e cor dos vértices e arestas, tipo de visualização e quais informações serão exibidas.

3.7 Versão Web

A versão web utiliza módulo CGI Python para coletar as informações fornecidas pelo usuário e repassá-las ao programa.

4 Principais desafios enfrentados

A Falta de padronização existente, atualmente, na Plataforma Lattes, exigiu a criação de procedimentos que permitam contornar esse problema. Em particular, os seguintes problemas são tratados:

– Falta de padronização dos nomes completos dos pesquisadores.
– Falta de padronização nos endereços profissionais.
– Visualização de árvores complexas correspondentes a genealogia acadêmica de pesquisadores com grande impacto na formação de mestres e doutores.

5 Resultados

Aqui apresentamos alguns resultados obtidos. Veja também nas seguintes páginas web alguns exemplos de geração de árvores de genealogia acadêmicas: http://www.linux.ime.usp.br/~edsonkm/mac499/ ou http://www.vision.ime.usp.br/creativision/genealogiaLattes/ .

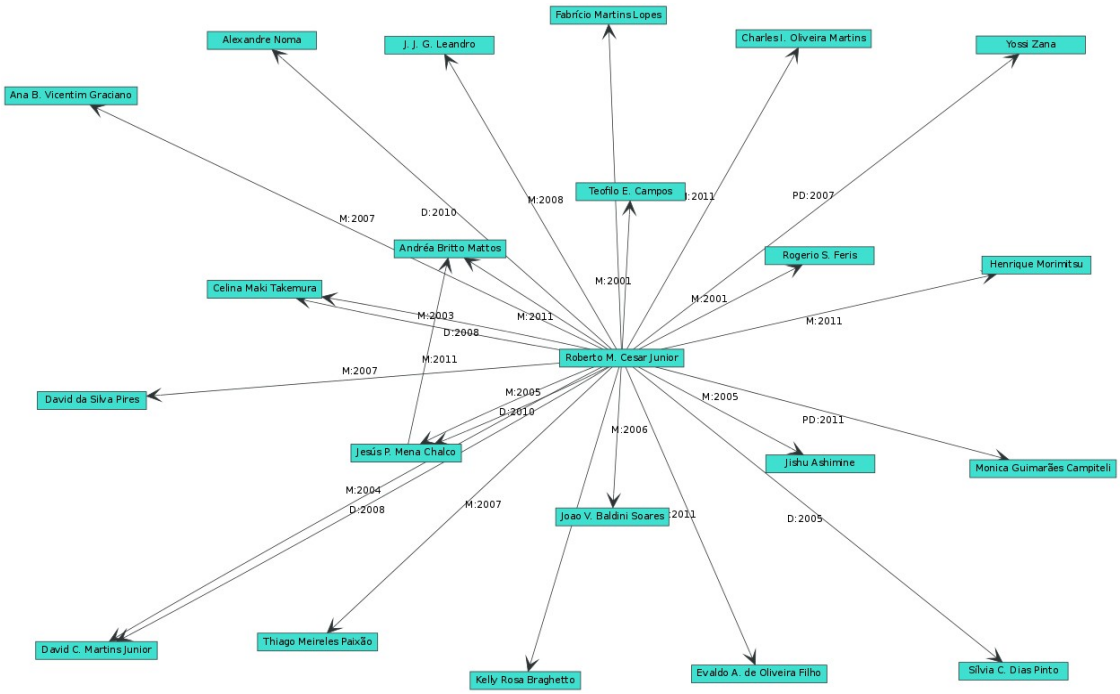

Figura 6: Árvore de genealogia acadêmica descendente do Prof. Dr. Roberto M. Cesar Jr.

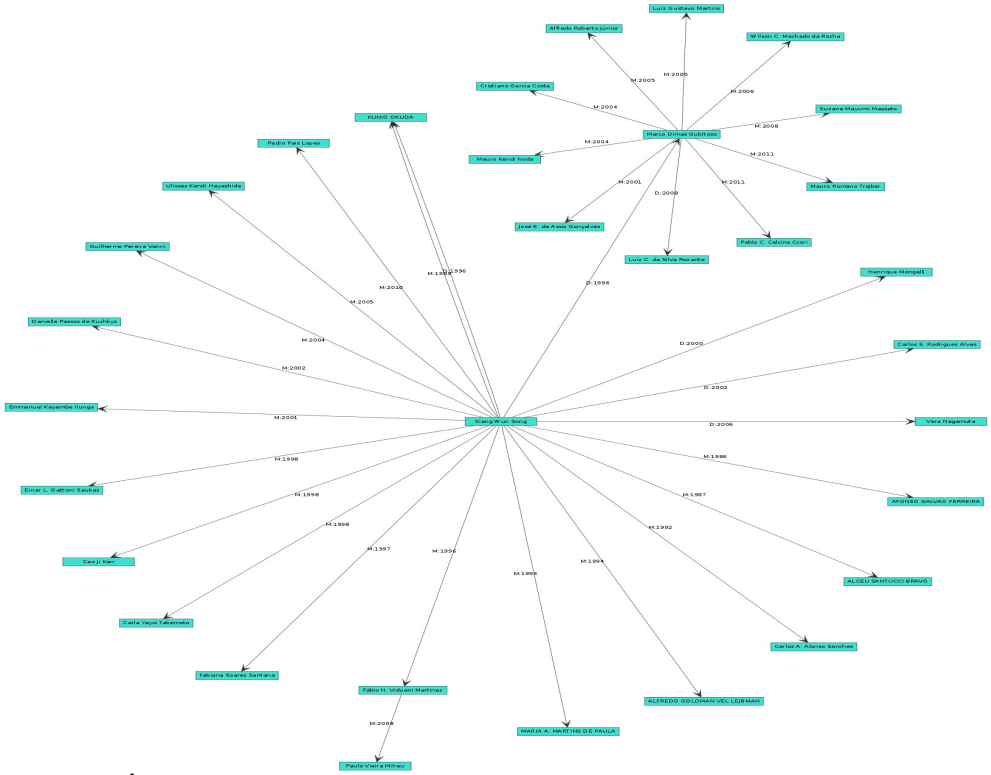

Figura 7: Árvore de genealogia acadêmica descente do Prof. Dr. Siang Wun Song.

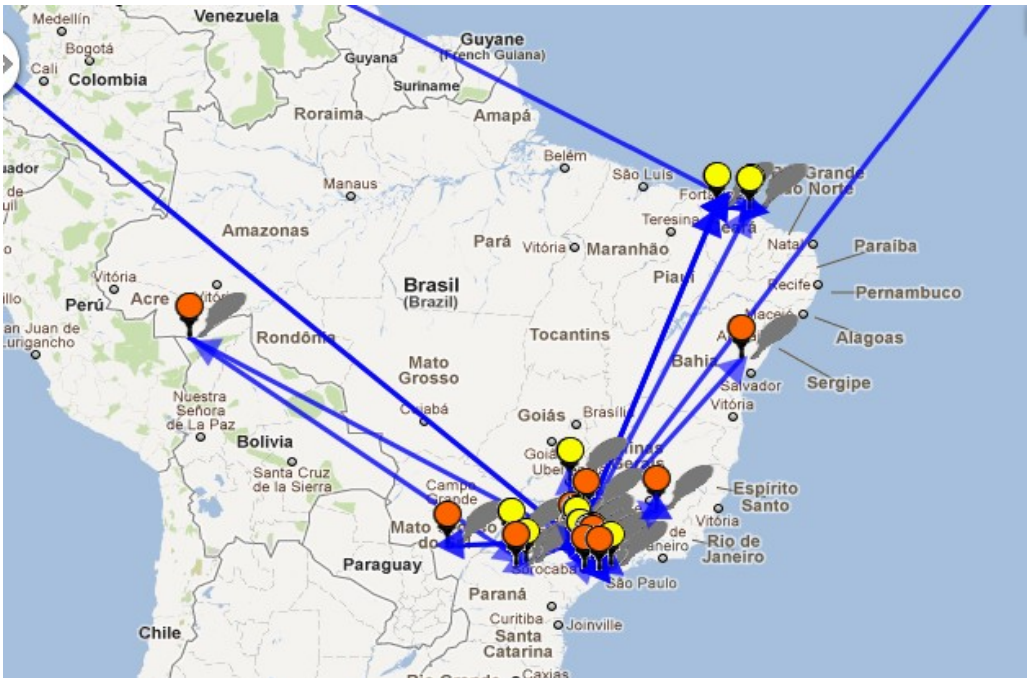

Figura 8: Representação geográfica da árvore de genealogia acadêmica descendente do Prof. Dr. Luciano da Fontoura Costa.

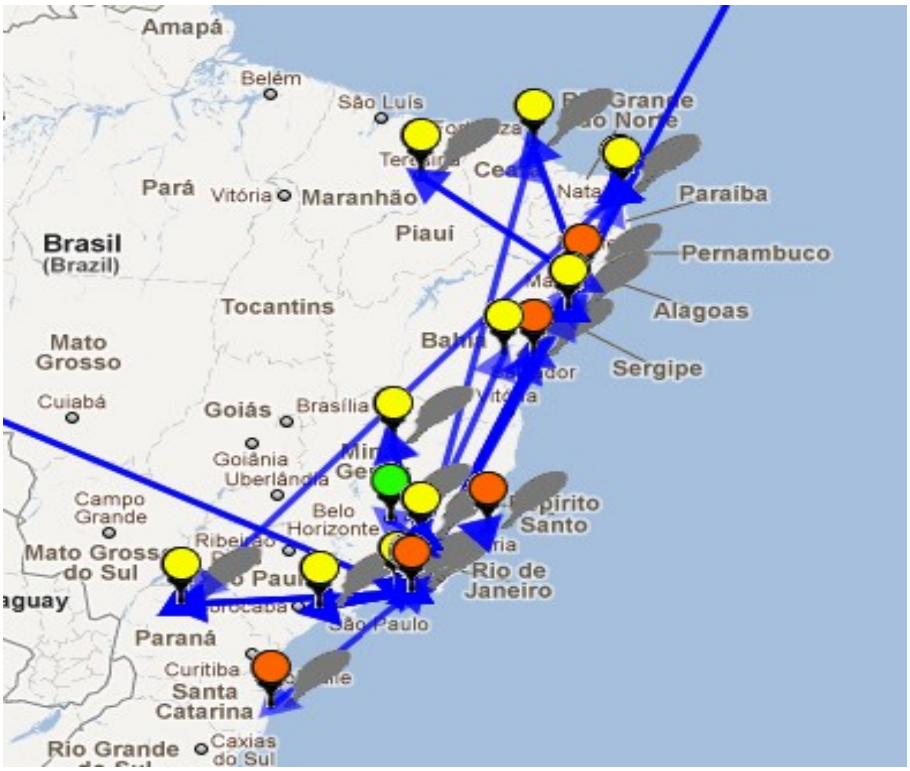

Figura 9: Representação geográfica da árvore de genealogia acadêmica descendente do Prof. Dr. Constantino Tsallis.

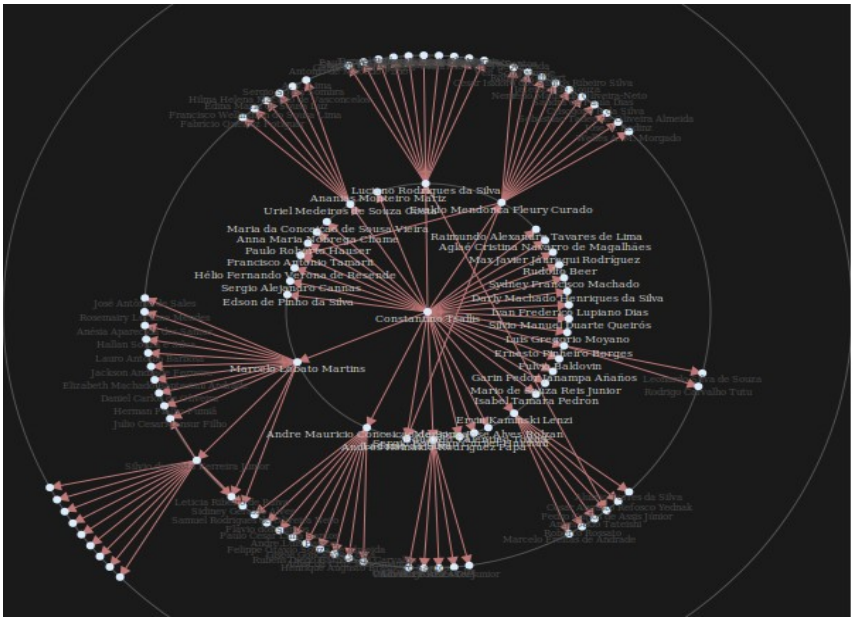

Figura 10: Representação interativa da árvore de genealogia acadêmica descendente do Prof. Dr. Constantino Tsallis, gerada por uma ferramenta externa (InfoVis Toolkit) usando os dados salvos em formato JSON.

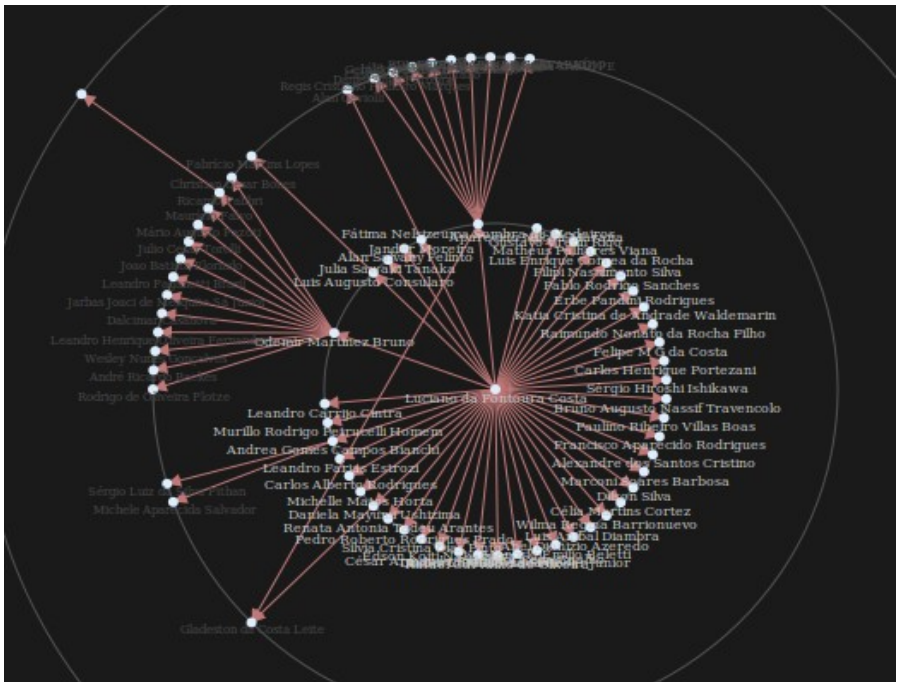

Figura 11: Representação interativa da árvore de genealogia acadêmica descendente do Prof. Dr. Luciano da Fontoura Costa, gerada por uma ferramenta externa (InfoVis Toolkit) usando os dados salvos em formato JSON.

6 Conclusão e trabalhos futuros

Este trabalho permite criar de forma automática árvores de genealogia acadêmica de usuários cadastrados na Plataforma Lattes. Foram criados vários procedimentos para obter diretamente da Plataforma Lattes todas as relações de orientação (concluída)

As árvores de genealogia acadêmicas geradas são salvas em formato PNG, GraphML e JSON. O formato PNG é uma imagem estática que fornece uma visão global, mas gera problemas com árvores muito grandes, pois a imagem se torna muito grandes, o formato GraphML guarda os dados da árvore utilizando a sintaxe XML, e o JSON utiliza estruturas do javascript, é um formato bastante simples e fácil de entender.

A representação geográfica da árvore, mostra a distribuição geográfica dos pesquisadores e acadêmicos, mas apresenta um limitação, pois só pode

mostrar as pessoas que estejam cadastradas na Plataforma Lattes, também apresenta um problema, se os pesquisadores ou acadêmicos trabalham num mesmo local as marcações no mapa estarão sobrepostos.

O trabalho também utiliza uma ferramenta externa, InfoVis Toolkit, que a partir dos dados salvos no aquivo de formato JSON gera uma visualização interativa da árvore, que se mostrou interessante principalmente na visualização de árvores de genealogia grandes.

Como trabalho futuro podem ser consideradas medidas/métricas que permitam caracterizar diferentes grafos correspondentes aos árvores de genealogia. Dessa forma, pode ser medido o impacto/influência de formação acadêmica de recursos humanos no Brasil. Características como grau do nó, caminho mínimo, e cliques podem ser explorados para árvores de genealogia (grafos direcionados) de todos os pesquisadores cadastrados na Plataforma Lattes.

7 Parte subjetiva

O desenvolvimento deste trabalho foi uma grande oportunidade para aprender um pouco sobre prospecção de dados, um área da inteligência artificial, que até então não possuía nenhum conhecimento prévio sobre o assunto. Mas no decorrer do desenvolvimento deste trabalho, demonstrou-se ser uma área bastante interessante e importante.

Com o uso de técnicas de prospecção de dados é possível obter conhecimentos como os que estão sendo usados por várias empresas de comércio *online* para inferir sobre os hábitos de consumo dos seus usuários.

Foi interessante ver como o parser extrai as informações dos currículos Lattes que estão em formato HTML para então podermos ter acesso a essa informações.

Aqui apresento algumas disciplinas que foram importantes no desenvolvimento desse projeto.

MAC110 – Introdução a Programação

Foi uma disciplina importante, pois foi onde comecei a estudar uma linguagem de programação mais a fundo.

MAC122 – Princípios de Desenvolvimento de Algoritmos

Aqui foi onde entrei em contato com estruturas de dados básicas e a como desenvolver algoritmos usando essas estruturas.

MAC323 – Estruturas de Dados

É uma disciplina muito importante, foi onde aprendi a usar e manipular diversas estruturas de dados.

MAC328 – Algoritmos em Grafos

Essa disciplina foi onde aprendi sobre as estruturas para representar grafos e diversos conceitos e algoritmos que atuam sobre grafo, que são muito importante na resolução de diversos problemas.

Referências

[1] Google Maps API Family: http://code.google.com/intl/pt-BR/apis/maps/index.html

[2] J. P. Mena-Chalco e R. M. Cesar-Jr. Prospecção de dados acadêmicos de currículos Lattes através de scriptLattes. Capítulo do livro Bibliometria e Cientometria: reflexões teóricas e interfaces (in press). São Carlos: Pedro & João, páginas 1-20, 2011.

[3] Mathematics Genealogy Project: Mission Statement. Disponível em http://genealogy.math.ndsu.nodak.edu/mission.php

[4] Plataforna Lattes: Histórico. Disponível em http://lattes.cnpq.br/conteudo/historico.htm

[5] ScriptLattes: An open-source knowledge extraction system from the lattes platform. Journal of the Brazilian Computer Society, v. 15, n. 4, p. 31-39, 2009.

[6] Xu, G.; Zhang, Y.; Li, L. Web Mining and Social Networking: Techniques and Applications, p. 5-7, Springer, 2010.