UNIVERSIDADE ABERTA

O algoritmo PageRank aplicado a redes de recomendações para seleção de recursos humanos

David Augusto da Silva Paiva Fernandes

Mestrado em Tecnologias e Sistemas Informáticos *WEB*

2018

UNIVERSIDADE ABERTA

O algoritmo PageRank aplicado a redes de recomendações para seleção de recursos humanos

David Augusto da Silva Paiva Fernandes

Mestrado em Tecnologias e Sistemas Informáticos *WEB*

Dissertação orientada pelo Professor Doutor

Luís Manuel Pereira Sales Cavique Santos

2018

RESUMO

Um dos principais desafios encontrados num processo de recrutamento de recursos humanos prende-se com a dificuldade prática de analisar de forma objetiva todos os candidatos. A grande quantidade de *curricula vitae* normalmente envolvida num processo deste tipo dificulta, quando não impede mesmo, uma leitura completa e, portanto, uma análise detalhada.

Neste trabalho procura mostrar-se a aplicabilidade do algoritmo PageRank a redes de recomendações profissionais entre pares no âmbito de processos de recrutamento de recursos humanos.

Assim, através de técnicas de *data-mining*, procedeu-se à extracção de dados de teste do conteúdo HTML de páginas de alguns profissionais registados no *site* LinkedIn, concretamente as recomendações efectuadas por cada um para um conjunto de outros profissionais por especialidade.

Com os dados extraídos contruiram-se redes de recomendações por especialidade, em que os vértices são os profissionais analisados e os arcos são as relações de recomendação, redes sobre as quais se aplicou o algoritmo PageRank para classificação ordenada de cada profissional por especialidade.

A extensão para várias especialidades é realizada através de um algoritmo de avaliação multicritério que, aplicado aos valores de PageRank, possibilita a classificação de um profissional tendo em conta mais do que uma especialidade.

Palavras chave: recursos humanos, rede de recomendações, PageRank, multicritério.

ABSTRACT

One of the main challenges encountered in a human resource recruitment process is the practical difficulty of analysing all candidates in an objective manner. The large amount of curricula vitae normally involved prevents a thorough reading and therefore a detailed analysis.

This work aims to show the applicability of link-analysis techniques to networks of professional recommendations among peers in the scope of recruitment processes of human resources.

So, using data-minign techniques, we proceed with the extraction of test data from the HTML content of some LinkedIn pages, moreover the recommendations made between members.

This extraction allowed the construction of a network of recommendations by skill, in which the nodes are professionals and the arcs are the recommendarions. Over these networks we applied the PageRank Algorithm in order to serialize each professional by skill.

The expansion to more than one skill is made through the use of an algorithm of multi-criteria evaluation, applied to the various PageRank values of each skill.

Keywords: human resources, recommendations network, multi-criteria, PageRank

DEDICATÓRIA

Dedico este trabalho à Patrícia, à Beatriz e à Inês, pelo carinho que me dispensam e orgulho que me alimentam, e por não me deixarem desistir, sem sequer uma queixa do que isso naturalmente implica.

AGRADECIMENTOS

À minha família em primeiro lugar, presente e partida, próxima e afastada, em especial aos meus pais José e Celeste, pela solidez da certeza de quem sou e pela garantia de lugares para onde há sempre caminho.

Aos meus professores, todos, pelo exemplo e motivação constante, e que por tantos serem aqui personifico nas pessoas do professor José Coelho, primeiro contacto nesta caminhada iniciada em 2009 na Licenciatura em Informática nesta mesma Universidade, da professora Gracinda Carvalho, orientadora do projeto final da licenciatura e de uma forma muito especial, do orientador desta dissertação de mestrado, professor Luís Cavique, pela paciência, conselho sempre atento e pertinente e pelo constante desafio e liberdade, já desde o tempo da licenciatura.

À Universidade Aberta, instituição pública de ensino superior a distância, e a todos os que fazem dela um corpo vivo e atuante, pela possibilidade que dá a todos os que, como eu, perseguem este sonho sem possibilidade de lhe dedicar um tempo regular e regulado. É uma instituição imprescindível e insubstituível.

Aos muitos colegas de percurso, apoio e irmandade nas dificuldades e nos sucessos, que por impossibilidade de nomear um a um, a todos reúno na pessoa, amigo, do João Rodrigues.

Obrigado.

Índice

RESUMO v

ABSTRACT vi

DEDICATÓRIA vii

AGRADECIMENTOS viii

Índice ix

Índice de tabelas xi

Índice de figuras xii

1 Introdução 3

1.1 Recrutamento em Recursos Humanos 3

1.2 Motivação e objetivos do trabalho 5

1.3 Contributos 8

1.4 Organização do documento 8

2 Trabalho relacionado e informação de base 13

2.1 A tecnologia ao serviço do recrutamento e seleção 13

2.2 PageRank em sistemas de recomendação 15

2.3 Redes e grafos 15

2.4 Análise de ligações *(link-analysis)* 17

2.5 Análise de decisão multi-critério (*multiple criteria decision analysis - MCDA*) .. 20

3 Extração de dados 25

3.1 Rede de recomendações 26

3.2 Acesso aos dados 29
3.3 O processo de extração de dados 30
4 Modelo proposto em R 39
4.1 Packages utilizadas 39
4.2 Rotinas implementadas 41
5 Resultados e Visualização 49
5.1 Redes com 1 atributo 50
5.2 Redes com 2 ou mais atributos (MCDA) 55
6 Conclusões e trabalhos futuros 65
6.1 Trabalhos futuros 65
Bibliografia 67
Anexos 73
A – Código fonte 75

Índice de tabelas

Tabela 2.1 - Exemplos de casos reais de aplicação de grafos 16

Tabela 5.1 - Especialidades 49

Tabela 5.2 - Candidatos para SQL com melhor PR 51

Tabela 5.3 - Candidatos para Java com melhor *page-rank* 55

Tabela 5.4 - 20 melhores candidatos em Java 56

Tabela 5.5 - 20 melhores candidatos em C# 56

Tabela 5.6 - 20 melhores candidatos em Sql 57

Tabela 5.7- 20 melhores candidatos em MySql 57

Tabela 5.8 - Tabela de candidatos com conhecimentos de sql, mysql, csharp e java 58

Índice de figuras

Figura 1.1 - Quadro de recomendações na página de um membro do LinkedIn 7

Figura 2.1 - Atividades de *link-analysis* (Samatova et. al, 2014) 17

Figura 3.1 - Perfil básico do membro 26

Figura 3.2 - Quadro de recomendações na página de um membro 29

Figura 3.3 - Arquitetura do sistema de extração e armazenamento de dados 31

Figura 3.4 - Autores das aprovações da área SQL Server 32

Figura 3.5 - Conteúdo HTML do diálogo de aprovações 33

Figura 3.6 - Modelo de dados (aprovação) 35

Figura 5.1 - Grafo da especialidade SQL 51

Figura 5.2 - Detalhe A (/in/jose/) 52

Figura 5.3 - Detalhe B (/in/david/) 52

Figura 5.4 - Grafo da especialidade Java 53

Figura 5.5 - Detalhe C (/in/david/) 54

Figura 5.6 - Detalhe D (/in/antonio/) 54

Capítulo 1

Introdução

1 Introdução

Neste capítulo apresentamos o problema da seleção de recursos humanos bem como a motivação, objetivos da dissertação e seus contributos para a sua mitigação.

1.1 Recrutamento em Recursos Humanos

A Oracle (2013) estima que o impacto financeiro do recrutamento de um colaborador de alto potencial (um *top performer*) é 10 a 100 vezes superior ao seu vencimento o que, refere ainda, numa conjuntura de grande concorrência, incerteza económica e elevados custos, é crucial para o sucesso da organização.

A procura de talento é pois uma grande preocupação das organizações, tanto privadas como públicas (Tsugawa e III, 2004; Ban, 2006), e um tema que merece grande atenção e congrega importantes investimentos em processos RH (Recursos Humanos) de recrutamento e seleção. A Oracle (2013) calcula que um total combinado de US$ 85.000.000.000 são gastos globalmente, anualmente, em processos de recrutamento e Bersin (2012) estima que um total de US$ 130.000.000.000 serão gastos em *software*, serviços, conteúdos e consultoria. Parece evidente que qualquer ganho nestes processos pode ter um enorme impacto em termos de tempo e custo.

1.1.1 Processo de recrutamento e seleção

Um processo de recrutamento começa por compreender a identificação da necessidade, a descrição da função a desempenhar e por definir o perfil psicoprofissional da função (Oliveira, 2010). Colocado em marcha o processo de busca do talento, o processo típico de recrutamento passa por diversas fases: (Gusdorf et al., 2008):

- **candidatura**, fase em que os candidatos são normalmente convidados a apresentar o seu curriculum e a preencher um formulário de candidatura, através do qual dão a conhecer as suas características e qualificações;
- **entrevista de triagem**, normalmente telefónica, em que, através de algumas perguntas diretas se procura determinar a adequação das qualificações do candidato à posição que se pretende preencher;

- **testes de seleção** com os quais se pretende identificar as competências do candidato que não puderam ser aferidas na entrevista de triagem e permitirão classificar os candidatos em termos de: aptidão, personalidade, competências, honestidade e motivação;

- **entrevista presencial**, individual ou pela equipa, em que num ambiente menos determinista do que um teste escrito, se procura conhecer a pessoa e aferir de que forma o candidato se comportará no trabalho;

- **verificação das referências** em que, após selecionado o candidato, se verifica a informação fornecida bem como a veracidade das referências apresentadas pelo candidato;

- **oferta da posição**, última fase do processo na qual se efetiva a oferta de trabalho e na qual se acertam os últimos detalhes como sejam o salário e outras condições.

A maioria das organizações realiza ainda um último passo, mais tarde, e que trata da avaliação do processo de recrutamento em termos de custo, tempo despendido e adequação do novo colaborador à posição ocupada.

1.1.2 Reputação na recomendação

Um dos principais problemas levantados pelo elevado número de CV que, com frequência, são recebidos num processo de recrutamento, prende-se com a verificação, em tempo útil, de competências, referências e percurso profissional. Gusdorf et al. (2008) refere estudos que indicam que, nos Estados Unidos, 40% dos candidatos mentem sobre o seu percurso profissional e académico e estima-se que 30% dos candidatos colocam algum tipo de falsidade nos seus CV's. Um inquérito a alunos universitários refere que 95% dos inquiridos afirmam que mentiriam para conseguir trabalho e 41% reconhecem já o ter feito em algum momento. Num outro inquérito, este efetuado a gestores de topo, conclui-se que 15% dos entrevistados admitem ter falsificado o seu CV.

Neste contexto, o conceito de recomendação aliado ao de reputação (Gkorou et al., 2015), podem ter um papel importante na classificação de elementos numa rede de relações e, no

caso particular do recrutamento, podem constituir um indicador muito útil na avaliação de um potencial colaborador.

Reputação

O site de partilha de conhecimento Stack Overflow (http://stackoverflow.com/) já referido, é uma plataforma que permite de forma gratuita a colocação de dúvidas e questões e reúne ajudas à sua solução. O sistema permite que se pontuem (como prémio) as respostas recebidas, no entanto, esta possibilidade de atribuir pontos a uma resposta carece de um determinado nível reputacional do membro pontuador e tal nível é obtido através da interação anterior do membro, seja através de respostas colocadas a outras questões seja por participação em discussões.

Otsuka et al. (2012) apresentam um método de avaliação da reputação na rede social Facebook com recurso a algoritmos que levam em linha de conta a topologia e estrutura da rede (Samatova et al., 2014) e os padrões nem sempre evidentes (Kubica et al., 2003), mais do que dos conteúdos observados. São disso exemplo o HITS (Li et al., 2002), capaz de descrever a estrutura de ligações entre número de comentários e relações de amizade e o PageRank (Brin e Page, 1998), seminal na classificação de grandes volumes de informação baseado apenas na estrutura de relações entre os elementos de uma rede, no caso, para efeitos de ordenação por "relevância" de páginas *WEB*.

Actualmente, a utilização de uma recomendação num processo de recrutamento está limitada ao contexto trivial de referência direta, isolada e individual de alguém. A sua validade está associada à reputação de quem a faz, normalmente reconhecida porque conhecido do próprio avaliador. É portanto uma utilização discreta e limitada ao conhecimento do avaliador. Num âmbito isolado de avaliação de apenas um profissional, faz sentido; num âmbito mais alargado de avaliação de múltiplos candidatos, torna-se impraticável.

1.2 Motivação e objetivos do trabalho

A motivação do trabalho é a de explorar um método alternativo, mais rápido e eficaz, de classificação e seriação de candidatos a uma dada necessidade de contratação profissional

recorrendo a dados do LinkedIn que envolvem recomendação dos pares. O que propomos é a aplicação do algoritmo PageRank a redes de recomendações, que não carece do conhecimento de qualquer tipo de informação que caracterize o candidato (cuja análise é precisamente a causa da lentidão dos processos actualmente em uso) mas antes centra a sua classificação na análise da rede de recomendações entre pares.

A técnica proposta permite inferir um indicador de competência de um dado profissional através da análise da rede de recomendações em que se insere, sem que seja necessário ao avaliador recorrer aos seus conhecimentos concretos de cada um dos elementos da rede. Tanto os indicadores de reputação de quem recomenda como da competência de quem é recomendado são calculados com base na estrutura da própria rede.

1.2.1 Caso de uso

Embora o LinkedIn não divulgue todos os métodos e algoritmos que utiliza, sabemos que, primeiro, a correspondência entre uma oportunidade/necessidade e um candidato se faz, sobretudo, com recurso à análise dos conteúdos disponiblizados por este na sua página e, segundo, a seriação dos vários candidatos possíveis, é feita através de métricas também relacionadas com esses conteúdos nomeadamente, o grau de compleição do perfil como seja o número e o tipo de campos preenchidos: se disponibilizou ou não anteriores colocações, se tem disponível o seu percurso académico, etc.

Exemplo:

Existindo a necessidade de recrutar um profissional com competência em *s* especialidades, de um conjunto de *n* candidatos, quais os *p* candidatos com melhor perfil para o desempenho da função?

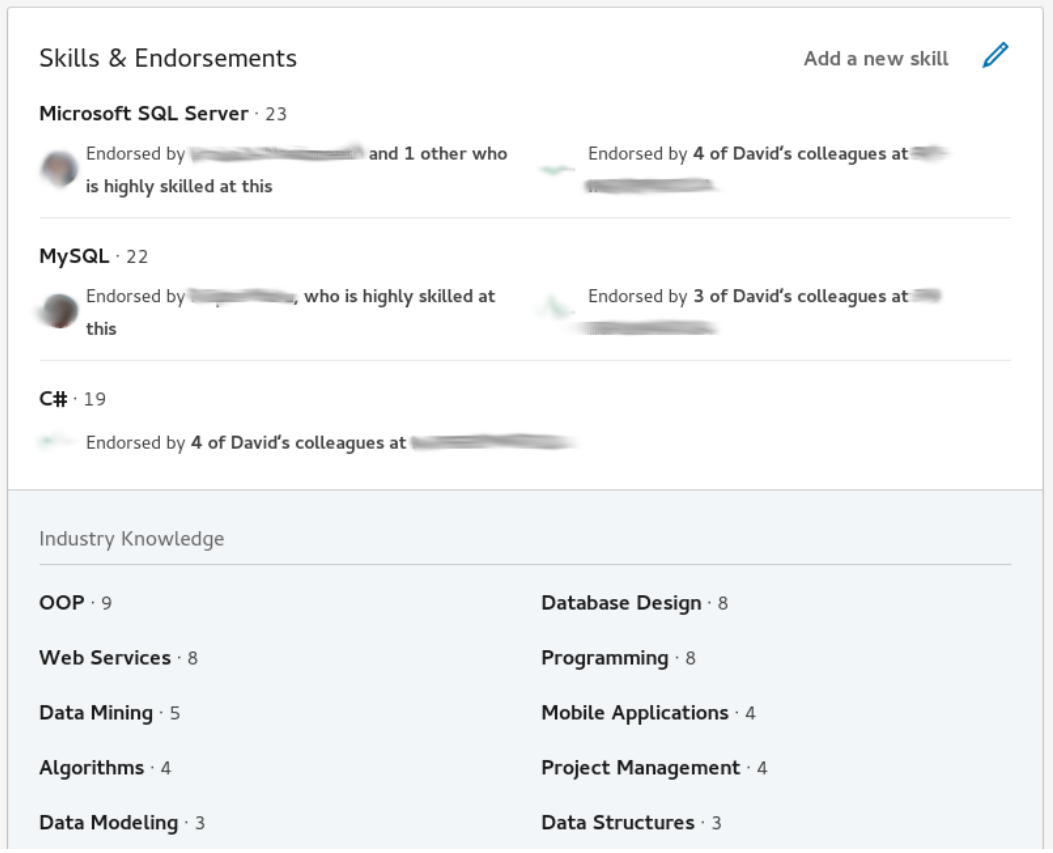

Figura 1.1 - Quadro de recomendações na página de um membro do LinkedIn

Uma empresa que se dedica ao desenvolvimento de *software* precisa de contratar 1 profissional com competências nas áreas da programação em Java e bases de dados *Sql*. Esta empresa possui *currivula vitae* (CV) de todos os candidatos bem como acesso às recomendações que cada um deles recebeu de pares e que se encontram registadas num sistema do tipo do LinkedIn.

O departamento de recursos humanos da empresa não tem capacidade para, em tempo útil, analisar detalhadamente cada um dos CV pelo que pretende fazer uma triagem inicial de candidatos com base nas referidas recomendações.

Objetivos a alcançar:

A definição do perfil do candidato é uma das componentes do processo de recrutamento e normalmente é constituída por competências num conjunto de especialidades, com pesos potencialmente distintos.

A avaliação de um candidato consiste, portanto, na valorização do seu desempenho em cada uma das especilidades.

Nesta investigação, como nos centramos apenas na rede de recomendações, propomos utilizar uma técnica de análise de ligações (*link-analysis*) mais especificamente o algoritmo PageRank (Brin e Page, 1998) aplicado às redes de recomendações que extraimos do LinkedIn.

A conjugação dos valores de PageRank de várias especialidades de um profissional é feita com recurso ao algoritmo TOPSIS (Hwang e Yoon, 1981) que consiste num método de avaliação multiatributo.

Descrevemos com mais detalhe ambos os algoritmos nas secções 2.4.4 e 2.5.3 respetivamente.

1.3 Contributos

Demonstramos neste trabalho a possibilidade de utilização de redes de recomendações como forma de seriar candidatos a uma determinada necessidade profissional para um conjunto de competências, potencialmente conflituantes; apresentamos uma proposta de modelo de dados de armazenamento deste tipo de redes, bem como propomos dois algoritmos de avaliação, apresentando casos concretos de utilização com o respectivo código fonte.

Mostramos ainda o processo de extracção automática de dados de teste a partir de páginas HTML da rede LinkedIn, dados esses utilizados nos testes realizados e visualizações produzidas.

1.4 Organização do documento

Este trabalho desenvolve-se em três fases:

1. Extracção de dados da rede social LinkedIn
2. Ordenação dos candidatos utilizando o algoritmo PageRank;
3. Para escolha de múltiplas competências é utilizado multi-critério TOPSIS.

Além deste capítulo inicial de introdução ao problema, apresentação da motivação, proposta de solução e contributos, a dissertação encontra-se dividida em 5 capítulos que a seguir se descrevem sumariamente.

No Capítulo 2 faz-se a apresentação de trabalho relacionado bem como se procede ao enquadramento teórico necessário à compreensão dos vários conceitos relacionados com o problema e sua solução: uma breve introdução aos grafos, algoritmo PageRank e análise de decisão multi-critério;

Embora a dissertação se proponha apresentar uma solução de certo modo genérica baseada numa rede de recomendações qualquer não específica, houve necessidade de se obter dados que pudessem ser utilizados nos testes efetuados e nos algoritmos escolhidos. Assim, procedeu-se a uma extração limitada de dados de recomendações de páginas do LinkedIn, através de um processo de HTML *data-mining* que se descreve no Capítulo 3.

Toda a análise de dados foi efetuada em R através da utilização de packages: igraph e MCDA, e no Capítulo 4 descrevem-se as funções utilizadas bem como rotinas desenvolvidas.

No Capítulo 5 são apresentados os resultados obtidos assim como diversas visualizações gráficas e numéricas dos mesmos.

Conclui-se esta dissertação com a apresentação de conclusões e propostas de desenvolvimento futuro no Capítulo 6, bibliografia consultada e termina com o código fonte das rotinas de extração de dados referidas no Capítulo 3.

Capítulo 2

Trabalho relacionado e informação de base

2 Trabalho relacionado e informação de base

Neste capítulo referimos o trabalho relacionado: em 2.1 a tecnologia ao serviço do recrutamento e selecçã e em 2.2 o algoritmo PageRank em sistemas de recomendação, bem como a informação de base para a leitura do artigo: em 2.3 redes e grafos, em 2.4 análise de ligações, e em 2.5 análise de decisão multi-critério.

2.1 A tecnologia ao serviço do recrutamento e seleção

Cada uma das fases do processo de recrutamento e seleção pode fazer uso de sistemas tecnológicos que possibilitam reduzir custos, acelerar processos e melhorar a qualidade dos resultados obtidos (Karaa e Amdouni, 2011).

Utilização das redes sociais

A internet veio alterar radicalmente os processos de gestão de recursos humanos tradicionais, principalmente como meio de comunicação (Kessler et al., 2009). De facto, através do potencial de comunicação da internet, empregadores e candidatos ficaram mais próximos e consequentemente viram agilizados todos os processos de recrutamento, seleção e colocação.

Esta facilidade de comunicação teve como resultado o aparecimento de *sites* especializados na recolha e armazenamento de CV, disponibilizando ferramentas de pesquisa aos empregadores que procuram profissionais para as suas empresas.

Capiluppi et al. (2013) referem um estudo de Barnes e Mattson (2009) que conclui que 48% das companhias da lista Inc. 500[1] utilizaram *sites* e redes sociais para recrutamento e avaliação de candidatos e um inquérito, efetuado a 115 PME's, em que se verificou que 78% das empresas inquiridas utilizaram também redes sociais nos seus esforços de recrutamento. Os mesmos autores indicam ainda as três redes sociais mais utilizadas: LinkedIn, Facebook e Jobster.

[1] Revista Inc. : http://www.inc.com/

Sistemas

O crescimento da Internet veio facilitar a comunicação entre empregadores e candidatos o que implicou um aumento exponencial no número de candidaturas a uma determinada oferta de trabalho. Tal volume de informação trouxe a necessidade da criação de sistemas que possibilitassem às empresas efetuar o tratamento da informação em tempo útil, nomeadamente a análise de CV.

Inúmeros sistemas surgiram para apoio ao mercado do recrutamento, como sejam: seguimento e avaliação de candidatos, sistemas de gestão de relacionamento de candidatos, sistemas de referência social e ferramentas de entrevista, mas o tipo de sistema mais interessante neste mercado são as ferramentas de deteção de recursos (Bersin, 2012) e sua classificação (Kessler et al., 2008).

Tecnologias

Uma parte significativa do trabalho realizado num processo de recrutamento, passa pela análise de textos escritos, desde logo os CV, e também de testes realizados. Kessler et al. (2009) elencam os diferentes tipos de necessidades e as correspondentes abordagens, desde o reconhecimento automático de formatos de CV, ao reconhecimento do seu conteúdo, através de identificação de termos específicos que possibilitem uma categorização.

Strohmeier e Piazza (2013), de uma forma mais abrangente, analisam as potencialidades do *data-mining* neste tipo de problema e referem como áreas de utilização: a seleção de candidatos, a previsão do retorno do colaborador, o recrutamento e deteção de competências, planeamento de carreiras, desenvolvimento e planeamento de custos de RH, previsão e avaliação do desempenho do colaborador.

Estes são problemas em que a disciplina de Inteligência Artificial desempenha um importante papel e o mesmo estudo apresenta métodos de *data-mining* utilizados: árvores de decisão, análise de *clusters* e de associações, *support vector machines* e redes neuronais.

Bakar e Ting (2011) aplicam redes Bayesianas a um problema com que se deparam os recrutadores na elaboração dos requisitos para uma contratação: a escolha das competências transversais não técnicas, as chamadas soft skills, mais adequadas a uma determinada necessidade de recrutamento e Domeniconi et al. (2016) usam análise

semântica latente (Dumais, 2005), uma técnica de processamento de linguagem natural, para encontrar semelhança semântica entre as competências de membros do LinkedIn.

2.2 PageRank em sistemas de recomendação

Esta investigação tem a sua origem no desejo de aprofundar um trabalho exploratório realizado no âmbito da componente curricular deste mestrado (Fernandes, 2015) e apresentado na InforAberta 2015, Jornadas de Informática da Universidade Aberta. Neste trabalho introduz-se a utilização do algoritmo *PageRank* a uma rede de recomendações extraída da informação sobre *Skills/Endorsements* do LinkedIn. Neste trabalho propõe-se ainda o desenvolvimento de um método que permita a dedução de informação adicional através da análise de correlações entre especialidades.

O algoritmo PageRank aplicado a redes de recomendações é também utilizado por Pérez-Rosés et al. (2016) e Pérez-Rosés e Sebé (2017). Nestes artigos descrevem a dedução de um indicador de autoridade de recomendação baseado na aplicação de um algoritmo de ordenação como o PageRank. Previamente à utilização do PageRank, os autores enriquecem informação contida no grafo de uma dada especialidade com novos arcos de peso ponderado através do cálculo de correlações entre diferentes especialidades.

Os autores propõem ainda uma metodologia de validação dos resultados que não dependa de forma acentudada do factor humano nem se apoie em informação privada dos candidatos.

2.3 Redes e grafos

São inúmeras as situações em que um conjunto de elementos se relacionam entre si. Em linguagem natural é comum utilizar-se o termo "rede" para identificar tais situações.

Newman (2003) apresenta vários exemplos de redes:

- **redes sociais**: conjuntos ou grupos de elementos que partilham algum tipo de contacto ou interação entre si: amizade entre pessoas, relações de negócio entre empresas, casamentos entre famílias;

- **redes de informação**: também chamadas redes de conhecimento como por exemplo: a rede de citações entre artigos académicos, a World Wide Web (WWW);
- **redes tecnológicas**: apresentadas como redes naturais ou construídas pelo homem de modo a distribuir algum tipo de bem ou recurso como seja a eletricidade. Alguns exemplos são: a rede elétrica e a rede telefónica; nos transportes, a rede de estradas ou ferroviária e as rotas da aviação comercial; na natureza os rios e afluentes;
- **redes biológicas**: desde logo a cadeia alimentar cujos elementos são espécies animais e vegetais e as suas relações de predação, as redes neuronais, seus neurónios e sinapses, a rede vascular suas veias e artérias.

Outros exemplos de redes: os amigos de escola, a árvore genealógica da família, a rede de contatos profissionais, a expansão de uma epidemia infecciosa, a vizinhança de países.

Biggs et al. (1986) definem **grafo** como sendo um conjunto finito de **vértices**, um conjunto finito de **arestas** e uma regra que nos indica que arestas **unem** que vértices.

É simples identificar quais os elementos e quais as regras de alguns dos exemplos referidos acima:

Tabela 2.1 - Exemplos de casos reais de aplicação de grafos

Caso	Elementos	Regra
Rede de estradas	Cidades	É possível chegar por estrada da cidade A até à cidade B diretamente sem passar por nenhuma outra cidade
Rede ferroviária	Estações	A estação B sucede à estação A
Amigos de escola	Pessoas	O João e o Manuel foram colegas de escola
Árvore genealógica	Pessoas	O Luís é pai da Joana
Contactos profissionais	Pessoas	A Rute trabalha/trabalhou com a Teresa
Expansão de uma epidemia	Pessoas	O David infetou a Patrícia
Vizinhança de países	Países	França e Espanha partilham fronteira

Neste trabalho utilizamos dados retirados das recomendações da rede social LinkedIn.

2.4 Análise de ligações *(link-analysis)*

Através da análise de ligações, procura extrair-se conhecimento de estruturas do tipo grafo. Estas análises combinam características da área da extração de dados (*data-mining*) e da topologia.

Um grafo pode ser homogéneo se os seus vértices representam elementos de um mesmo tipo ou heterogéneo no caso contrário. Um exemplo de um grafo homogéneo é a WWW em que os vértices representam endereços *WEB*. Um exemplo de um grafo heterogéneo é o que representa uma rede de citações bibliográficas na qual os vértices representam diferentes tipos de obras: livros, artigos, revistas, etc. (Samatova et al., 2014).

São várias as aplicações de *link-analysis* a problemas reais, desde as redes de computadores (que servidores são passíveis de receber mais carga de trabalho) à WWW (que páginas são mais relevantes dentro de um determinado assunto).

Tendo em conta o resultado pretendido, há duas perspetivas principais de análise:

- classificação (*link-based object classification-LOC*) e
- ordenação (*link-based object ranking-LOR*).

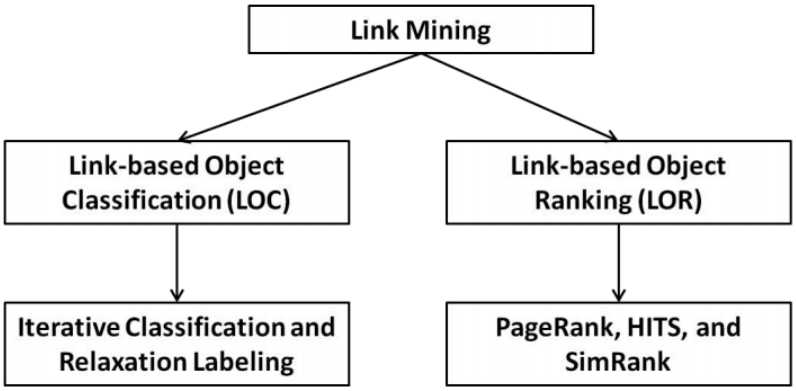

Figura 2.1 - Atividades de *link-analysis* (Samatova et. al, 2014)

2.4.1 Classificação (*Link-based Object Classification*)

Esta técnica visa unicamente agrupar vértices de uma rede de acordo com determinadas propriedades topológicas ou características. Samatova et. al (2014) apresentam diversos exemplos:

- classificar os vértices de uma rede de acordo com a sua conectividade, forte ou fraca, recorrendo ao seu grau;
- calcular a distância média de cada vértice a todos os outros e classificá-los segundo essa medida. Assumindo uma rede representada por um grafo completo, tal classificação permitiria conhecer o grau de centralidade de cada vértice.

A classificação pode ainda recorrer a propriedades ou atributos dos elementos da rede, por exemplo a idade ou o género dos membros de uma rede social, e combinar esses dados com informação de tipo estrutural, topológico.

2.4.2 Ordenação (*Link-based Object Ranking*)

A ordenação de objetos baseada em ligações procura atribuir um grau de importância aos vértices da rede baseado na estrutura de ligações. Enquanto as técnicas de classificação atribuem aos vértices etiquetas de um conjunto finito de possibilidades, as técnicas de ordenação procuram atribuir uma valorização relativa a cada um dos vértices.

Uma das mais conhecidas técnicas de ordenação baseada em ligações é aquela que atribui graus de relevância a páginas *WEB* (Brin e Page, 1998) tendo sido demonstrado que a relevância de uma determinada página *WEB* está relacionada, não apenas com o seu conteúdo, mas também com as ligações (no caso, hiperligações) entre ela e outras páginas *WEB*.

2.4.3 Predição de ligação

Uma terceira tarefa prende-se com a predição de mutação de uma determinada rede baseada na análise das alterações ocorridas na sua estrutura. Por exemplo, qual vai ser a necessidade de computadores na empresa ou de novas estradas na rede viária do distrito baseando-se na análise das alterações da estrutura operadas nos últimos anos.

2.4.4 PageRank

O algoritmo conhecido como PageRank (Brin e Page, 1998) responde a um problema de *Link-based Object Ranking* (LOR), no caso, especificamente aplicado a páginas *WEB*. A ideia passa por calcular um indicador numérico (o valor de PageRank da página) indicativo da relevância de uma página WWW explorando apenas a estrutura de ligações da *WEB* e em especial a quantidade e qualidade das ligações que lhe são feitas (Samatova et al., 2014).

Sendo a WWW uma rede, facilmente se percebe a associação, em estrutura, a um grafo no qual os vértices são as páginas e as arestas as hiperligações entre elas.

Da mesma forma, ressalta aqui a semelhança em termos de estrutura entre a rede de páginas WWW e uma rede de recomendações, razão pela qual nos parece um algoritmo aplicável, também, no nosso problema.

Ao calcular a relevância de um vértice, o algoritmo PageRank, iterativo, leva em linha de conta o grau de entrada desse vértice e o valor de PageRank dos vértices cujas arestas o apontam, o referenciam, o recomendam.

Matematicamente, o grafo pode ser representado por uma matriz estocástica em que cada valor representa, de facto, a probabilidade de um navegador aleatório passar por a página (no caso da WWW) ou pelo profissional (no nosso caso) que esse valor representa:

$$PR(i) = (1-d) + \sum_{j:\, i \text{ aponta para } j} PR(j)\frac{d}{C(j)}, i = 1, \dots, N \qquad (2.1)$$

em que $C(j)$ é o número de páginas para que aponta a página j , isto é, o grau de saída do vértice que a representa.

Neste trabalho iremos utilizar, para ordenação dos candidatos, o algoritmo PageRank.

2.5 Análise de decisão multi-critério (*multiple criteria decision analysis - MCDA*)

Segundo Belton e Stewart (2002), num contexto de tomada de decisão é necessário observar um critério, um *standard*, pelo qual se possa julgar uma decisão ou uma tomada de ação como sendo preferível a outra.

A consideração de diferentes alternativas ou cursos de ação torna-se num problema de tomada de decisão com múltiplos critérios (*multiple criteria decision making - MCDM*) quando existe um conjunto de critérios, de *standards*, em conflito.

2.5.1 Otimalidade de Pareto e curvas de indiferença

O economista italiano Vilfredo Pareto (1848-1923) foi quem primeiro estudou matematicamente a agregação de critérios conflituantes num único índice composto e foi ainda quem introduziu o conceito de eficiência, conhecida como **otimalidade de Pareto**, conceito chave em economia e na moderna teoria da MCDM.

Köksalan et al. (2013) referem que uma alocação de recursos **ótima à Pareto** é atingida quando não é possível melhorar a situação de alguém sem piorar a situação de outrem. Esta ideia aplicava-se originalmente ao problema da negociação entre várias pessoas mas é generalizável para problemas de uma só pessoa e múltiplos critérios.

Francis Edgeworth (1845-1926) introduziu o conceito de **curva de indiferença** como sendo o gráfico de uma função que representa a combinação de variáveis à qual um agente é indiferente na medida em que qualquer que seja a combinação considerada o nível de satisfação, também referida como utilidade, providenciado não se altera.

2.5.2 Modelo de preferência

Em face de um problema de decisão, devem ser identificadas as alternativas disponíveis e um conjunto de critérios contra os quais as alternativas serão avaliadas e comparadas.

As alternativas podem ser identificadas sob a forma de uma lista discreta ou sob a forma de um conjunto de constrangimentos ou vetor de variáveis de decisão.

Para se poder efetuar a avaliação e a comparação de alternativas, é necessário construir um modelo que seja capaz de representar as preferências e as avaliações. Um **modelo preferencial** deve conter dois componentes principais:

1. Preferências em termos de **critérios individuais** que representem a importância relativa de se atingirem diferentes níveis de desempenho para cada critério identificado;
2. Um **modelo de agregação** que permita efetuar comparações inter-critério tais como compromissos (*trade-off).*

2.5.3 TOPSIS

O algoritmo TOPSIS (Hwang e Yoon, 1981) (*Technique for order preference by similarity to ideal solution*), é uma técnica de avaliação multi-atributo que se baseia no princípio de que, de um conjunto de observações de vários atributos, aquela que se constituirá como preferencial será a que se encontre à menor distância euclidiana da **solução ideal positiva** e à maior distância euclidiana da **solução ideal negativa**. Assume-se que os atributos podem tomar valores de utilidade monotonicamente crescentes ou decrescentes.

Os valores dos atributos são representados através de uma matriz bidimensional M_{mn} com *n* colunas, uma por atributo, e *m* linhas, uma por observação.

Algoritmo TOPIS:

1. **Normalização da matriz de decisão**, operação que possibilita a comparação de atributos uma vez que transforma os valores dos atributos em valores não dimensionais:

$$\boldsymbol{r_{ij}} = \frac{\boldsymbol{x_{ij}}}{\sqrt{\sum_{k=1}^{m} \boldsymbol{x_{kj}^2}}} \qquad (2.1)$$

2. Construção da matriz normalizada de pesos, que se obtém multiplicando a matriz de decisão *M* por um vetor de pesos $w = (w_1, w_2, \dots, w_n)$ estocástico ($\sum_{j=1}^{n} w_j = 1$) que deve ser fornecido pelo decisor.
3. Determinar as soluções ideais positiva A^* e negativa A^-:

$$A^* = \{(\max_i v_{ij} | j \in J), (\min_i v_{ij} | j \in J') | i = 1,2, \dots m\}$$
$$= \{\boldsymbol{v_1^*}, \boldsymbol{v_2^*}, \dots, \boldsymbol{v_n^*}\} \qquad (2.2)$$

$$A^- = \{(\min_i v_{ij} | j \in J), (\max_i v_{ij} | j \in J') | i = 1,2, \dots m\}$$
$$= \{\boldsymbol{v_1^-}, \boldsymbol{v_2^-}, \dots, \boldsymbol{v_n^-}\} \quad (2.3)$$

Em que:

$J = \{j = 1,2, \dots, n \mid j \text{ associado a critérios com benefício}\}$

$J' = \{j = 1,2, \dots, n \mid j \text{ associado a critérios com custo}\}$

4. Calcular a medida de separação, que mais não é do que a distância euclidiana, entre cada uma das alternativas e cada uma das soluções ideais:

$$\boldsymbol{S_{i*} = \sqrt{\sum_{j=1}^{n} (v_{ij} - v_j^*)^2}, i = 1, 2, \dots m} \quad (2.4)$$

e

$$\boldsymbol{S_{i-} = \sqrt{\sum_{j=1}^{n} (v_{ij} - v_j^-)^2}, i = 1, 2, \dots m} \quad (2.5)$$

5. Calcular a proximidade relativa à solução ideal:

$$C_{i*} = \frac{S_{i-}}{S_{i*} + S_{i-}}, 0 < C_{i*} < 1, \ i = 1,2, \dots m \quad (2.6)$$

Obviamente que uma alternativa A_i estará tanto mais próxima da solução ideal A^* quanto mais próximo C_{i*} estiver de 1.

6. Ordenação por ordem de preferência:

Um conjunto de alternativas pode então ser ordenado por ordem decrescente de C_{i*}.

Neste trabalho, para escolha de candidatos com multiplas competências, recorremos ao algoritmo de análise multi-critério TOPSIS.

Capítulo 3

Extração de dados

3 Extração de dados

O LinkedIn é a “maior rede profissional do mundo com centenas de milhões de membros, cuja missão consiste em conectar profissionais por forma a que sejam mais produtivos e bem sucedidos”. (LinkedIn, 2016)

Entre as várias potencialidades anunciadas, como sejam a criação do perfil profissional, estabelecimento de contacto com colegas de trabalho ou de escola, a organização afirma poder ajudar o profissional a:

1. Construir e manter a sua rede profissional;
2. Encontrar outros profissionais da mesma indústria;

Estas são as duas características que mais interessam para este trabalho já que contêm em si as características de rede e pesquisa.

O modelo de dados que permite caracterizar um membro do LinkedIn é bastante completo e nem todos os atributos e estruturas interessam para esta investigação. Apresentamos no entanto o modelo e alguns dos atributos registados já que poderá ser útil para trabalhos futuros.

Parte da informação que caracteriza um membro consiste em atributos alfanuméricos como sejam: nome, fotografia, contactos, curriculum, entre muitos outros, e um outra parte consiste em informação relacional: as recomendações escritas por colegas, clientes ou outro tipo de contacto. Na Figura 3.1 podemos observar os atributos disponíveis no denominado perfil básico do membro.

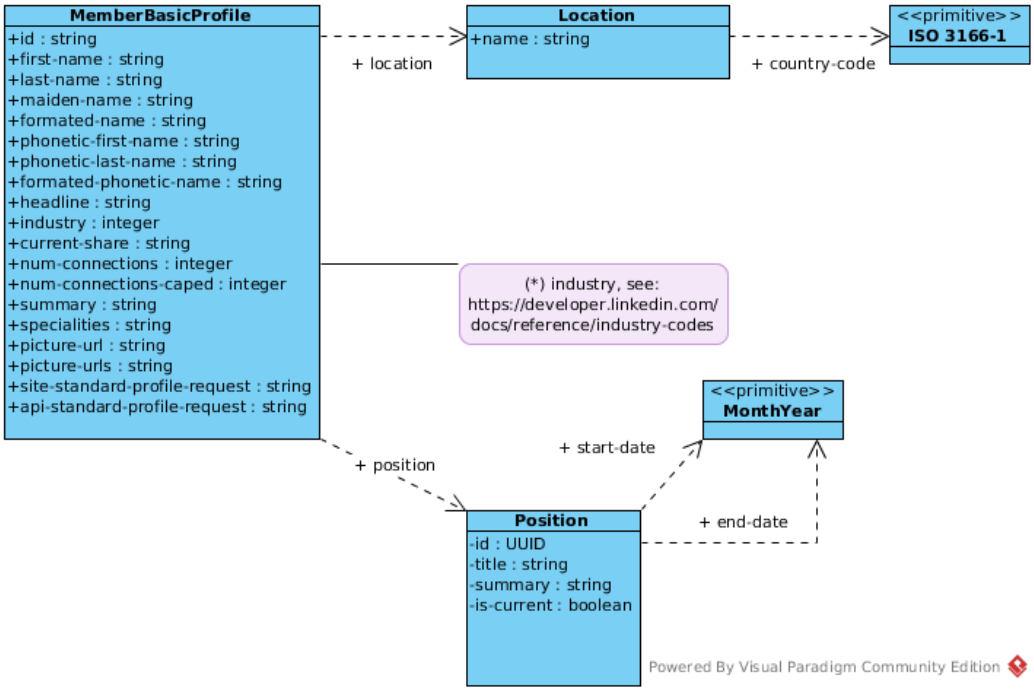

Figura 3.1 - Perfil básico do membro

A entidade principal é a chamada *MemberBasicProfile* (perfil básico de membro) que possui os atributos representados na Figura 3.1 além da ligação a duas outras entidades: ***Location***, que consiste no país do membro, e ***Position*** com consiste na ocupação do membro.

3.1 Rede de recomendações

A informação que nos interessa nesta investigação, e que infelizmente não é disponibilizada pela API do LinkedIn, é a lista de aprovações ou de competências reconhecidas. Estas aprovações não são mais do que referências de um membro às capacidades de um outro sobre um determinado assunto: o João indica que a Patrícia possui conhecimentos de Java, e o Manuel de Redes Neuronais.

De futuro, referiremos estas indicações como **rede de recomendações** que deve entender-se como tendo o seguinte significado: João recomenda a Patrícia como conhecedora de Java e recomenda o Manuel como entendido em Redes Neuronais.

Note-se que estas recomendações não possuem qualidade. Não se trata portanto de avaliações mas apenas referências positivas (não se podem fazer referências negativas) unárias: determinado membro refere outro como entendido numa determinada matéria. Na Figura 3.2 podemos ver a parte da página *WEB* de um membro na qual surge esta informação. Neste exemplo podemos ver que existem 15 membros que indicam o detentor desta página como conhecedor de Java, enquanto 19 o indicam como conhecedor de C#.

Importa notar que referimos o carácter unário (e não binário) destas recomendações na medida em que uma ausência de recomendação não indica que o membro não possua conhecimentos numa determinada área; apenas não foi assim indicado por aqueloutro membro mas tal não possui nenhum tipo de valor para esta investigação.

Já referimos o carácter de rede desta informação e importa precisar alguns das suas características à luz da teoria.

Antes de mais, e sem perda de generalidade, vamos desde já assumir que a cada especialidade: Java, Redes Neuronais, C#, etc., corresponderá uma rede própria autónoma.

3.1.1 Esta rede de recomendações é um digrafo

Sendo uma rede, pode ser representada por um grafo em que os vértices são membros registados no LinkedIn e as arestas relações de recomendação.

Por outro lado é um grafo sem **auto-laços**, já que um membro não pode recomendar-se a si próprio, assim como não podem ocorrer **arestas paralelas**: um membro não repete uma recomendação. Se tivéssemos optado por considerar um grafo único representando todas as especialidades, haveria naturalmente a ocorrência de arestas paralelas. Optamos por não o fazer.

O **grau de entrada** de um vértice corresponde ao número de membros que recomendam o membro representado por esse vértice para determinada especialidade; na Figura 3.2, os números que vemos ao lado de cada especialidade corresponde precisamente ao grau de entrada do vértice no grafo desse assunto.

Na medida em que são recomendações, trata-se portanto de um grafo **dirigido**; o facto de o membro A recomendar o membro B para um determinado assunto S, não significa que o membro B o faça em relação ao membro A. As arestas possuem portanto **orientação**.

No caso de se verificar esta recomendação mútua, isto é A recomendar B e B recomendar A, estamos na presença de um **laço orientado**, não apenas possível como natural de ocorrer.

Na medida em que não podem ocorrer **auto-laços** mas podem ocorrer **laços orientados**, não podemos garantir a aciclicidade do grafo.

3.1.2 Conectividade

É possível encontrar sequências de recomendações entre dois determinados membros, mas não garantidamente entre todos pelo que será possível encontrar **passeios abertos** e **fechados**, bem como **caminhos** e **circuitos**.

Não se pode garantir que seja um grafo conectado já que é possível pensar, por exemplo, duas recomendações entre 4 membros distintos da seguinte forma: A recomenda B, C recomenda D, o que configura um **grafo desconectado**, e também não se pode garantir que seja um **grafo completo** embora pouco provável tal não seja, de todo, impossível.

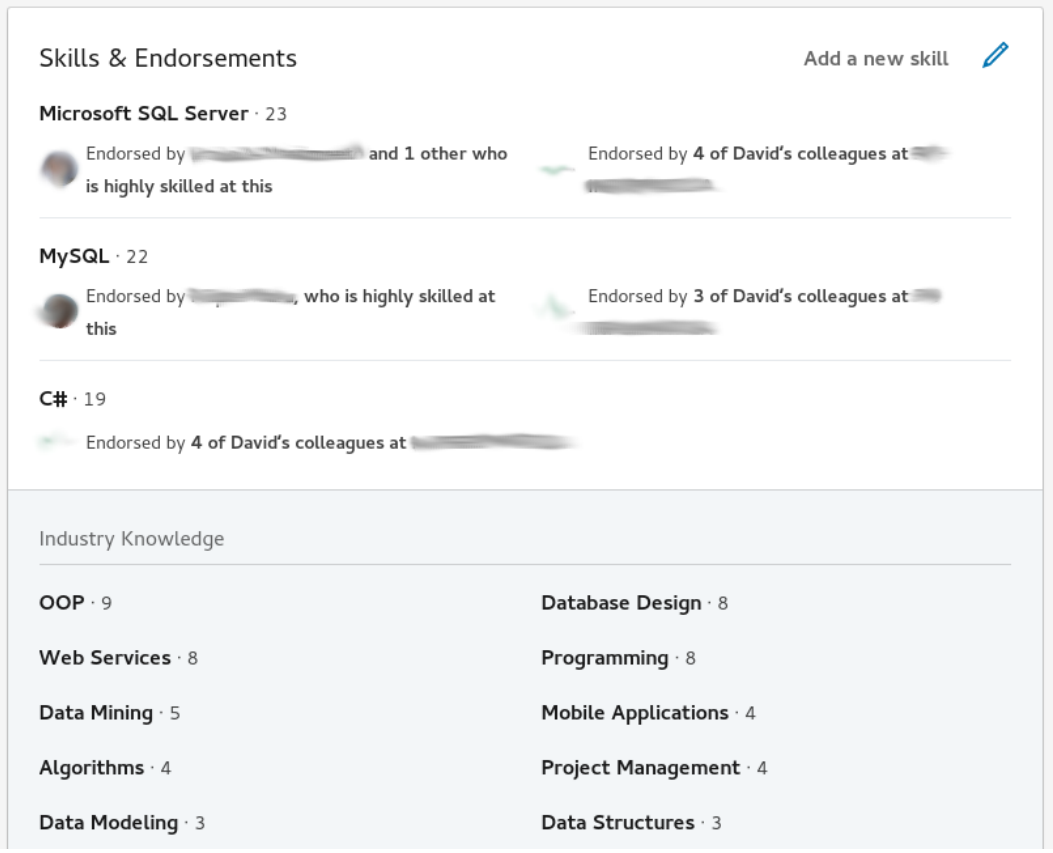

Figura 3.2 - Quadro de recomendações na página de um membro

3.2 Acesso aos dados

O LinkedIn disponibiliza uma API de acesso controlado que permite aceder a alguma da informação do membro, no entanto, como esta investigação se centra apenas na estrutura da rede de recomendações, informação que a API não disponibiliza, esta teve de ser extraída das páginas dos membros diretamente a partir do HTML.

O acesso ao conteúdo HTML de uma página está informalmente limitado de forma a que as operações de extração, sobretudo maciças, não interfiram no desempenho da página ou até dos serviços que a suportam. Tal é normalmente referido num ficheiro denominado `robots.txt` no qual se indicam o tipo de limitações para um conjunto de programas extratores conhecidos. Estas limitações visam controlar a exploração intensiva de conteúdos, nomeadamente a que é efetuada por motores de busca no trabalho de enriquecimento das suas bases de dados de indexação.

No caso desta investigação, a extração de dados, além de perseguir finalidades académicas sem nenhum outro intuito, limitou-se a um conjunto reduzido de dados, obtidos de forma apenas semiautomática e, porquanto, impassível de afetar ainda que de forma ligeira a operação do sítio do LinkedIn.

3.3 O processo de extração de dados

Incluído no contexto mais vasto da denominada "*web* aumentada" (Firmenich et. al, 2014) no qual se inclui, por exemplo, a personalização da navegação (Ankolekar e Vrandecic, 2008), os **scripts de utilizador** permitem a interação programática direta com o conteúdo HTML de uma página apresentada no navegador *WEB*. Esta interação é unicamente realizada no *browser* sem nenhum impacto nos serviços *WEB* servidores que geraram a página mas pode, não apenas aceder aos dados apresentados como alterá-los, eliminando, alterado ou adicionando conteúdo HTML, interagindo diretamente com o DOM (*document object model*) (W3C, 1998) da página em visualização.

É uma tecnologia que envolve questões relacionadas tanto com a segurança da navegação como com a autenticidade dos conteúdos e que conta, obrigatoriamente, com a necessidade de suporte dos programas de navegação.

A instalação de um destes scripts de utilizador é efetuada sempre por iniciativa do próprio utilizador não havendo forma de tal ser feito automaticamente sem o seu conhecimento.

Uma das primeiras iniciativas de implementação desta tecnologia foi a denominada GreaseMonkey (Comunidade, 2005) para o navegador Mozilla Firefox mas que desde então se alargou a praticamente todos os programas de navegação sob a forma de extensões.

A linguagem de programação destes scripts é a Javascript e a sua execução pode ocorrer antes (*beforeLoad*) ou depois (*afterLoad*) do carregamento da página. No caso da extração de dados interessa que a sua execução tome lugar depois da página totalmente carregada.

3.3.1 Arquitetura do sistema de extração

Por questões de segurança que facilmente se compreendem, apenas sob regras muito rígidas e condições bem definidas, um script executado no navegador tem acesso aos recursos locais do computador onde corre. Não é, por exemplo, possível aceder ao disco local de forma livre para gravar um ficheiro.

Na medida em que o processo de extração de dados, não totalmente automático, iria acarretar algum trabalho manual repetitivo, optou-se por criar um serviço WEB REST a ser invocado pelo script JS e que trataria de gravar a informação recebida numa pequena base de dados mySQL criada para o efeito.

A arquitetura deste sistema é a seguinte:

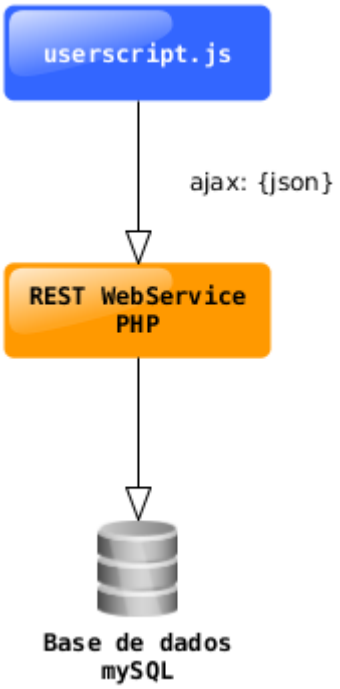

Figura 3.3 - Arquitetura do sistema de extração e armazenamento de dados

3.3.2 userscript.js

A Figura 3.2 mostra a parte da página de um membro que contém as especialidades recomendadas, referindo para cada uma o número de aprovações.

Clicando sobre uma especialidade, é possível consultar os membros que deram a sua aprovação, surgindo um diálogo do tipo do que se pode ver na Figura 3.4. No cabeçalho é indicada a especialidade bem como o número de aprovações efetuadas e, de seguida, surge a lista de membros que deram a sua aprovação, mostrando a fotografia, o nome e a função de cada um.

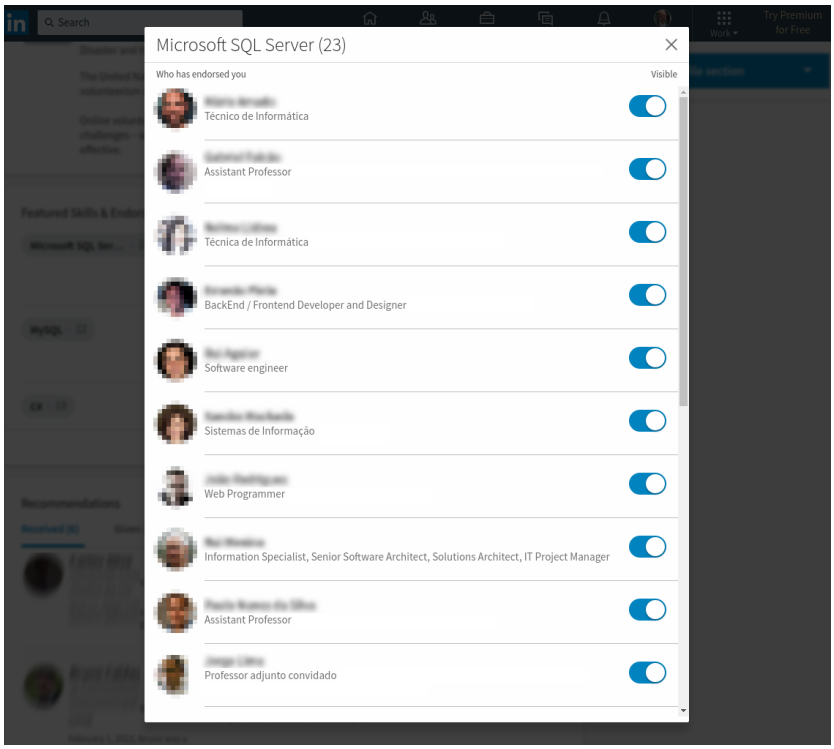

Figura 3.4 - Autores das aprovações da área SQL Server

Além da fotografia, nome e função, cada entrada de membro desta lista possui uma hiperligação para a página do próprio membro, informação essencial para a construção da rede de recomendações.

Na Figura 3.5 apresenta-se um extrato do conteúdo HTML subjacente a este diálogo salientando cada um dos elementos com informação relevante a ser extraída:

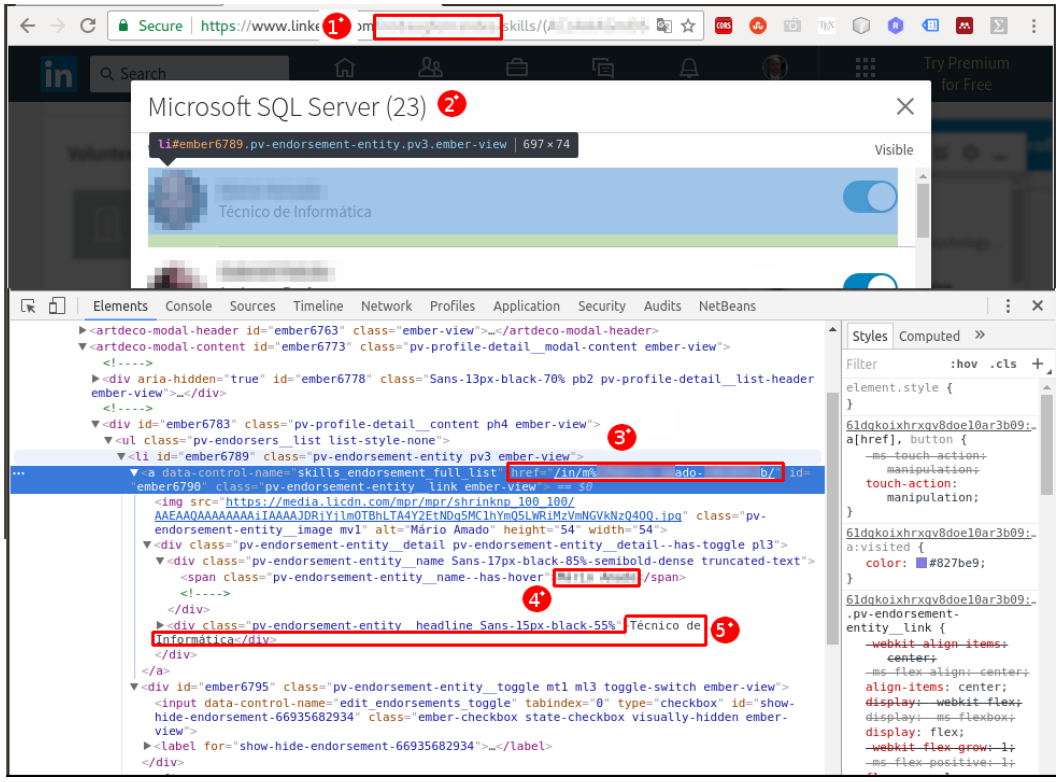

Figura 3.5 - Conteúdo HTML do diálogo de aprovações

1. URL da página de onde se pode extrair a identificação do membro recomendado;
2. Nome da especialidade de aprovação e número de membros aprovadores;
3. Hiperligação para a página do membro aprovador;
4. O seu nome
5. A sua função

A informação coletada é formatada numa estrutura JSON e enviada para o serviço WEB REST de armazenamento:

```
{
    "id": "/in/%id da página do membro aprovado_id%/",
    "area": "%área de aprovação%",
    "endorsers": [{
            "id": "/in/%id da página do membro aprovador_1%/",
            "name": "%nome do membro aprovador_1%"
        },
        ... ,
        {
            "id": "/in/%id da página do membro aprovador_n%/",
            "name": "%nome do membro aprovador_n%"

        }]
}
```

Código fonte 3.1 - Estrutura JSON de aprovações de um membro

Questões de implementação

Apresentam-se de seguida algumas questões que tiveram de ser observadas na implementação do script.

3.3.2.1.1 Início da extração

Partes do sítio do LinkedIn fazem uso de tecnologia *single page navigation* a qual, para evitar o carregamento da totalidade da página a cada interação, faz uso de chamadas AJAX (Casteleyn et. al., 2009, p.25) para obtenção de dados adicionais e, através de manipulação do DOM, completar informação ou alterar conteúdos.

Não era portanto suficiente fazer executar o **userscript.js** de extração apenas com o carregamento da página pelo que se implementou um ciclo infinito de deteção do aparecimento no DOM da estrutura HTML daquele diálogo e, uma vez detetada, processar o HTML e extrair a informação relevante.

3.3.2.1.2 Conteúdos de carregamento parcial

O diálogo da Figura 3.4 funciona por carregamento parcial de, no máximo, 20 membros. Nos casos em que o número de membros aprovadores é superior a 20, como o exemplificado (23), o carregamento dos restantes vai sendo efetuado à medida que o utilizador rola a lista até ao seu final.

Como se conhece o número total de membros esperados (no caso 23), o ciclo infinito de extração leva em linha de conta esse valor para detetar se já foram carregados todos os membros. Caso ainda não estejam carregada a totalidade dos membros esperada, o script provoca o rolamento automático da lista para o final e assim provocar o carregamento dos, eventualmente vários, lotes seguintes de membros.

3.3.2.1.3 Envio de informação coletada

O envio da informação coletada é efetuado através de uma chamada AJAX ao serviço *WEB* REST implementado. Uma vez que o domínio deste serviço (no caso *localhost*) não coincide com o domínio do *host* da página (www.linkedin.com) houve a necessidade de ultrapassar as limitações de controlo de acesso impostas pelo protocolo HTTP e que

impedem, por omissão, a partilha de recursos de origens cruzadas (*Cross-Origin Resource Sharing - CORS*) (Network, 2016).

3.3.3 REST WebService PHP

Com o objetivo de acelerar o processo de transferência da informação coletada pelo **userscript.js** para um local de onde pudesse ser posteriormente consumida, foi criado um serviço WEB REST em PHP sobre uma base de dados mySQL.

O modelo de dados muito simplificado permite o armazenamento da estrutura mostrada em Código fonte 3.1.

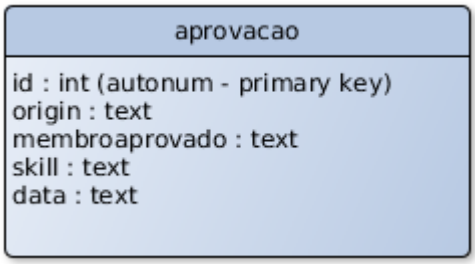

Figura 3.6 - Modelo de dados (aprovação)

- **id**: chave primária automaticamente gerada;
- **origin**: descrição da origem dos dados (informativo: userscript1, userscript2, introdução manual, etc.);
- **membroaprovado**: id LinkedIn do membro;
- **skill**: especialidade;
- **data**: conteúdo json das aprovações (a componente `endorsers` da estrutura json referida em Código Fonte 4.1).

Questões de implementação

Sendo um serviço muito simples, a única questão a observar foi a devida à limitação de partilha de recursos de origens cruzadas (CORS), como já referido, através da inclusão da

```
'Access-Control-Allow-Origin: *', true
```

Código fonte 3.2 - CORS

diretiva de cabeçalho HTTP:

Este serviço *WEB* identifica dois verbos: GET para obtenção dos dados e utilizado pela rotina R de cálculo e POST para inserção de dados coletados pelo **userscript.js** como já referido.

Estrutura de dados da rede de recomendações

```
{
   "id":"/in/daugfernandes/",
   "area":"Microsoft SQL Server",
   "endorsers":[
      {
         "id":"/in/mário-amado-14626912b/",
         "name":"Mário Amado"
      },
      {
         "id":"/in/gfalcao/",
         "name":"Gabriel Falcão"
      },
      {
         "id":"/in/nelmalisboa/",
         "name":"Nelma Lisboa"
      },
      {
         "id":"/in/ernesto-pinto-13742b88/",
         "name":"Ernesto Pinto"
      },
      ...,
      {
         "id":"/in/reisfrederico/",
         "name":"Frederico Reis"
      },
      {
         "id":"/in/ferreiran/",
         "name":"Nuno Ferreira"
      },
      {
         "id":"/in/paulonascimento/",
         "name":"Paulo A. Nascimento"
      }
   ]

}
```

Código Fonte 3.3 – Dados de aprovações de um membro

Definido o modelo de dados e extraida a informação de teste, iremos apresentar no capítulo seguinte as rotinas e packages R selecionadas, comentando cada uma das funções utilizadas.

Capítulo 4

Modelo Proposto em R

4 Modelo proposto em R

A componente de análise e visualização é efetuada com recurso a rotinas R.

4.1 Packages utilizadas

Apresentam-se de seguida as packages utilizadas, suas particularidades e pormenores de utilização.

4.1.1 igraph

A package igraph (Csardi e Nepusz, 2006) inclui várias funções para análise e visualização de redes e possui implementações em R (a que utilizamos), Python e C/C++. Nesta investigação utilizou-se a implementação do algoritmo PageRank (Brin e Page, 1998) disponibilizada na função **page_rank**:

```
1: page_rank(
2:    graph,
3:    algo = c("prpack", "arpack", "power"),
4:    vids = V(graph),
5:    directed = TRUE,
6:    damping = 0.85,
7:    personalized = NULL,
8:    weights = NULL,
9:    options = NULL)
```

Código fonte 4.1 - Função page_rank implementada pela package igraph

A função aceita 8 argumentos, nem todos obrigatórios:

- **graph**: um objeto do tipo `graph` com a definição do grafo de rede a analisar;
- **algo**: o algoritmo a utilizar no cálculo do valor de PageRank*:*
    - **prpack**: é a mais recente implementação, disponível a partir da versão 0.7 da package e, segundo os autores, a mais estável e rápida;

    - **arpack**: implementação disponível a partir da versão 0.5 que utiliza a livraria ARPACK;
    - **power**: o algoritmo original que utiliza o método das potências.
- **vids**: os vértices de interesse. Por omissão todos os vértices do grafo `graph`;
- **directed**: indica se o grafo é orientado, assim considerado por omissão;
- **damping**: o fator de damping d;
- **personalized**: vetor opcional com uma distribuição de probabilidades para cálculos personalizados, em que a probabilidade de abandono do passeio aleatório não seja uniforme. Se utilizado, este vetor deverá possuir uma entrada para cada vértice e será reescalado de forma a o tornar um vetor estocástico.
- **weights**: vetor opcional numérico que deve ser utilizado no caso de se pretender fornecer pesos personalizados às arestas. Se referido, substituirá eventuais pesos existentes como atributo das arestas do grafo.
- **options**: lista opcional de atributos do tipo nome/valor para substituir parametrizações por omissão dos algoritmos ARPACK ou POWER. Ignorado se utilizado o algoritmo PRPACK;
- **niter**: número máximo de iterações realizadas; 1000 por omissão no algoritmo POWER ;
- **eps**: diferença mínima entre o valor de PageRank de duas iterações sucessivas para se considerar o cálculo terminado; 0.001 por omissão no algoritmo POWER.

4.1.2 MCDA

A *package* MCDA (Meyer et. al, 2017) implementa um conjunto de funções de suporte à análise de decisão multi-critério.

TOPSIS

Esta package inclui a implementação do algoritmo TOPSIS (Hwang e Yoon, 1981) (*Technique for order preference by similarity to ideal solution*).

De notar que a algoritmo prevê que nem todos os atributos possuam a mesma importância. Para isso, permite a referência de um vetor de pesos, pesos esses que indicam numericamente a importância de cada um dos atributos a considerar.

A função TOPSIS aceita 7 argumentos:

```
1: TOPSIS(
2:    performanceTable,
3:    criteriaWeights,
4:    criteriaMinMax,
5:    positiveIdealSolutions = NULL,
6:    negativeIdealSolutions = NULL,
7:    alternativesIDs = NULL,
8:    criteriaIDs = NULL)
```

Código Fonte 4.2 - Função TOPIS implementada na package MCDA

- **performanceTable**: matriz de decisão com *n* colunas, tantas quantos os critérios, e *m* linhas, uma por alternativa;
- **criteriaWeights**: vetor de pesos dos critérios;
- **criteriaMinMax**: vetor de caracteres [“min”, “max”] que indica a direcção da preferência de cada critério. “min” indica que o critério deve ser minimizado e”max” que deve ser maximizado.
- **positiveIdealSolution**: vetor opcional com as solução ideal positiva;
- **negativeIdealSolution**: vetor opcional com as solução ideal negativa;
- **alternativesIDs**: opcional contendo os Ids das alternativas a considerar (se omisso serão consideradas todas);
- **criteriaIDs**: vetor opcional contentos os ID’s dos critérios a considerar (se omisso serão considerados todos).

4.2 Rotinas implementadas

De seguida apresentam-se as rotinas R implementadas para as várias fases do trabalho.

4.2.1 Leitura da base de dados e transformação de dados

A primeira fase refere-se à seleção de dados da base de dados de recomendações que foi criada através do WebService já referido. O código apresentado encontra-se comentado nos pontos mais importantes.

```
1:  select_data <- function() {
2:   con <- dbConnect(MySQL(),
3:                    user="guest", password="*********",
4:                    dbname="mestrado", host="localhost")
5:   on.exit(dbDisconnect(con))
6:   rs <- dbSendQuery(con, "select endorsed, skill, data from linkedin")
7:   data <- fetch(rs)
8:   void <- dbHasCompleted(rs)
9:   dbClearResult(rs)
10   return(data);
11: }
```

Após a leitura dos dados, que para este caso é efetuada sem nenhum filtro, uma série de transformações são necessárias de forma a adequar a estrutura de dados ao formato que a função *page_rank* exige.

```
1:  # selecina registos da base de dados
2:  data_rows <- select_data()
3:
4:  # concatenação do campo 'data' de todos os regsitos e forma a criar
5:  # um bloco JSON válido
6:  json_data_block <- paste(data_rows$data, collapse=",")
7:
8:  # criação de um objecto JSON
9:  json_data <- fromJSON(paste("[", stri_enc_toascii(json_data_block), "]",
collapse=""))
10:
11: # transformação da estrutura hierarquica da forma:
12: #    {member, endorsers[]}[]
13: # numa estrutura em série na forma
14: #    [{member, expertise, endorser}]
15: serial_data <- lapply(
16:   json_data,
17:   function(x)
18:   {
19:     lapply(
20:       x$endorsers,
21:       function(y) {
22:         c(x$id, x$area, y$id)  #y$id is endorser's id
23:       }
24:     )
25:   }
26: )
27:
28: # criação de uma matriz
29: serial_matrix <- t(matrix(nrow=3,unlist(serial_data)))
```

Código fonte 4.4 - Transformação de estrutura

4.2.2 Produção gráfica do grafo e cálculo do valor de PageRank

Obtidos os dados e previamente preparados através de transformações sucessivas, podem estes ser utilizados para a produção gráfica do grafo de recomendações bem como para cálculo dos valores de PageRank dos candidatos.

```
1:  # seleção de registos de uma especialidade
2:  somelinks <- function (dataframe, expertise='ANY') {
3:  subset <-
4:      if(expertise=='ANY') {dataframe }
5:      else { dataframe[dataframe$expertise==expertise,] }
6:  return(subset[,c("endorser","member")])
7:  }
8:
9:  # extrai id e nome dos membros recomendadores
10: # da estruturas {member, endorsers[]}[]
11: endorsers <- lapply(
12:  json_data,
13:  function(x)
14:  {
15:    lapply(
16:      x$endorsers,
17:      function(y) {
18:        c(y$id,y$name)
19:      }
20:    )
21:  }
22: )
23:
24: # extrai id e nome dos membros recomendados
25: # da estruturas {member, endorsers[]}[]
26: endorsees <- unique(lapply(
27:  json_data,
28:  function(x)
29:  {
30:      c(x$id,x$name)
31:  }
32: ))
33:
34: # cria matriz
35: endorsers_matrix <- t(matrix(nrow=2,unlist(endorsers)))
36: rownames(endorsers_matrix) <- endorsers_matrix[,1]
37: colnames(endorsers_matrix) <- c("Id","Nome")
38:
39: # criação de data.frame com nomes de colunas apropriados
40: dt <- unique(data.frame(serial_matrix))
41: colnames(dt) <- c("member","expertise","endorser")
42:
43: # obtem especialidades únicas
44: expertises <- sort(unique(dt$expertise))
45:
46: # obtém o grafo: endorser -> member (recomendador -> recomendado)
47: links <- data.frame(cbind(serial_matrix[,3],serial_matrix[,1]))
48:
49: # produz o objecto grafo
50: graph <- graph.data.frame(links)
```

Código fonte 4.5 - Produção do objeto graph

De seguida exemplifica-se a criação de um grafo para uma especialidade e produção da sua representação gráfica bem como da tabela de valores de PageRank.

```
1: # função utilitária de produção de PDF para uma determinada especialidade
2: doplot <- function(dataframe, expertise='ANY', folder, filename) {
3:   # clean names for members not endorsees
4:   links <- somelinks(dataframe, expertise=expertise)
5:   labels <- unlist(lapply(links$endorser,function(x){ if (x %in% links$member)
6:            {as.character(x)} else {''} }))
7:   print(labels)
8:   setwd(folder)
9:   pdf(filename, width = 30, height = 30)
10:  plot.igraph(graph.data.frame(links),
11:              main = expertise,
12:              vertex.label = unlist(links$X2),
13:              vertex.size = 5,
14:              rescale = TRUE,
15:              vertex.label.font=1,
16:              vertex.color="lightblue",
17:              edge.color="black")
18:  dev.off()
19: }
20:
21: # ===================================
22: # especialidade
23: exp <- "C#"
24:
25: gnet <- somelinks(dt,exp)
26: g <- graph.data.frame(gnet)
27: page_ranks <- data.frame(page.rank(g)$vector)
28: named_page_ranks <- cbind(rownames(page_ranks), page_ranks[,1])
29:
30: result <- named_page_ranks[order(named_page_ranks[,2],decreasing = T),]
31: colnames(result) <- c("member","PR")
32:
33: # faz merge com lista de membros para obter o nome
34: dt2 = merge(dt, result, "member")
35:
36: # produção de PDF com o grafo
37: doplot(dt2,  exp, "/home/david/documents", "csharp_net.pdf")
38:
39: # 20 melhores PR
40: csharp <- head(result,20)
```

Código fonte 4.6 - Produção de gráfico e tabela de PR

4.2.3 Análise multi-especialidade

Com o objetivo de efetuar a avaliação de candidatos tendo em conta mais do que uma especialidade, implementou-se o código que a seguir se descreve e que utiliza a função TOPSIS da *package* MCDA já referida anteriormente.

```
1:  # as variaveis seguintes: sql, mysql, csharp e java são obtidas
2:  # através do código anterior e são dataframes com dois campos: id do membro e PR
3:
4:  colnames(sql) <- c("member","sql")
5:  colnames(mysql) <- c("member","mysql")
6:  colnames(csharp) <- c("member","csharp")
7:  colnames(java) <- c("member","java")
8:
9:  # marge das várias tabelas numa única com os PR de todas as especialdiade
10: madf <- sql;
11: madf <- merge(mysql, madf, "member");
12: madf <- merge(csharp, madf, "member");
13: madf <- merge(java, madf, "member");
14:
15: # preparação dos dados para a função TOPSIS
16: row.names(madf) <- madf$member
17: madf <- madf[,c("sql","mysql","csharp","java")]
18:
19: madf$sql = as.numeric(as.character(madf$sql))
20: madf$mysql = as.numeric(as.character(madf$mysql))
21: madf$csharp = as.numeric(as.character(madf$csharp))
22: madf$java = as.numeric(as.character(madf$java))
23:
24: # escolha de pesos para cada um dos atributos, no caso indiferenciados
25: weights <- c(0.20,0.40,0.20,0.20);
26: names(weights) <- colnames(madf)
27:
28: # critérios min max a utilizar
29: criteriaMinMax <- c("max","min","max","max")
30: names(criteriaMinMax) <- colnames(madf)
31:
32: # especialidades a incluir
33: criteriaIDs <- c("csharp","sql","mysql")
34:
35: # resultado
36: result <- sort(
37:      TOPSIS(performanceTable =  madf,criteriaWeights = weights, criteriaMinMax =
38:        criteriaMinMax, criteriaIDs = criteriaIDs),
39:  decreasing = TRUE)[seq(4)]
```

Código fonte 4.7 - Análise multicritério

Deve notar-se que o código implementado foi produzido numa perspetiva de utilização em laboratório e não numa solução preparada para execução por utilizadores finais. Algum esforço de desenvolvimento será necessário para tornar a solução amigável ao utilizador final.

Com base nos calculos efectuados, iremos de seguida apresentar várias visualizações dos resultados, na forma gráfica e tabular.

Capítulo 5

Resultados e Visualização

5 Resultados e Visualização

Neste capítulo apresentam-se alguns resultados e respetivas visualizações das redes de recomendação de algumas especialidades.

Os dados de suporte utilizados nesta investigação foram obtidos, como foi referido no Capítulo 3, por extração automática de dados do LinkedIn, e correspondem a 6579 recomendações efetuadas a 15 membros por um total de 177 dos seus pares em 130 especialidades.

As análises efectuadas permitem ilustrar os objectivos do trabalho bem como salientar algumas das características dos algoritmos utilizados, nomeadamente, a quase imunidade do PageRank a recomendações falsas ou fraudulentas.

Tabela 5.1 - Especialidades

```
  [1,] "5S"
  [2,] "AJAX"
  [3,] "Algorithms"
  [4,] "Android"
  [5,] "Android
Development"
  [6,] "Android SDK"
  [7,] "AngularJS"
  [8,] "Application
Architecture"
  [9,] "Application
Management"
 [10,] "Architecture"
 [11,] "ASP"
 [12,] "ASP.NET"
 [13,] "C"
 [14,] "C  "
 [15,] "C#"
 [16,] "Cache"
 [17,] "Cascading Style
Sheets (CSS)"
 [18,] "C/C  "
 [19,] "Cloud Computing"
 [20,] "Cocoa"
 [21,] "Computer
Architecture"
 [22,] "Computer Science"
 [23,] "CSS"
 [24,] "CUDA"
 [25,] "Database Design"
 [26,] "Databases"
 [27,] "Data Center"
 [28,] "Data Mining"
 [29,] "Data Modeling"
 [30,] "Data Structures"
 [31,] "Digital
Electronics"
 [32,] "Eclipse"
 [33,] "Electrical
Engineering"
 [34,] "Embedded Systems"
 [35,] "Erlang"
 [36,] "FPGA"
 [37,] "French"
 [38,] "Geographic
Information Science"
 [39,] "GPGPU"
 [40,] "GPS"
 [41,] "High Performance
Computing"
 [42,] "HTML"
 [43,] "Hybrid Mobile
Development"
 [44,] "iOS"
 [45,] "iOS development"
 [46,] "iOS Development"
 [47,] "iPhone"
 [48,] "iPhone
development"
 [49,] "IT Management"
 [50,] "Java"
 [51,] "JavaScript"
 [52,] "jQuery"
 [53,] "jQuery UI"
 [54,] "Linux"
 [55,] "Location Based
Services"
 [56,] "Management"
 [57,] "Medical Imaging"
 [58,] "Microsoft Office"
 [59,] "Microsoft SQL
Server"
 [60,] "Mobile"
 [61,] "Mobile
Applications"
 [62,] "Mobile
Applications Development"
 [63,] "Mobile Devices"
 [64,] "MVC"
 [65,] "MySQL"
 [66,] "Navigation
Systems"
 [67,] ".NET"
 [68,] ".NET Framework"
 [69,] "Networking"
 [70,] "Objective-C"
 [71,] "Object Modelling"
 [72,] "Object-Oriented
Programming (OOP)"
 [73,] "OOAD"
 [74,] "OOP"
 [75,] "OpenCL"
 [76,] "Operating
Systems"
 [77,] "Optimization"
 [78,] "Oracle"
 [79,] "Parallel
Computing"
 [80,] "Perl"
 [81,] "PHP"
 [82,] "PL/SQL"
 [83,] "PostgreSQL"
 [84,] "Pro*C"
 [85,] "Product
Management"
 [86,] "Programming"
```

```
 [87,] "Project
Management"
 [88,] "Project Planning"
 [89,] "R&amp;amp;D"
 [90,] "Science"
 [91,] "Security"
 [92,] "Shell Scripting"
 [93,] "Signal
Processing"
 [94,] "SOAP"
 [95,] "Software
Analysis"
 [96,] "Software Design"
 [97,] "Software
Development"
 [98,] "Software
Engineering"
 [99,] "Software Project
Management"
[100,] "Solution
Architecture"
[101,] "SQL"
[102,] "SQL Server"
[103,] "Statistics"
[104,] "System
Administration"
[105,] "System
Architecture"
[106,] "Teaching"
[107,] "Team Leadership"
[108,]
"Telecommunications"
[109,] "Transact-SQL (T-
SQL)"
[110,] "T-SQL"
[111,] "UML"
[112,] "Unix"
[113,] "VB.NET"
[114,] "VHDL"
[115,] "Virtualization"
[116,] "Visual Basic"
[117,] "Visual Basic .NET
(VB.NET)"
[118,] "Visual Studio"
[119,] "VLSI"
[120,] "Web Applications"
[121,] "Web Development"
[122,] "Web Services"
[123,] "Windows"
[124,] "Windows 7"
[125,] "Windows Server"
[126,] "Windows XP"
[127,] "Xcode"
[128,] "XML"
[129,] "XSLT"
[130,] "Yii framework"
```

5.1 Redes com 1 atributo

Com o objectivo de mostrar graficamente algumas das características do algoritmo PageRank, nomeadamente a não estrita determinação à quantidade de arcos (recomendações), nesta secção apresentamos resultados de cálculos realizados apenas sobre uma especialidade. De facto, a representação gráfica das redes de múltiplos atributos não seria facilmente perceptível.

Mostramos assim redes de recomendações, bem como tabelas com os 6 candidatos com melhor PageRank, para algumas das especialidades.

5.1.1 Exemplo de recomendações da competência SQL

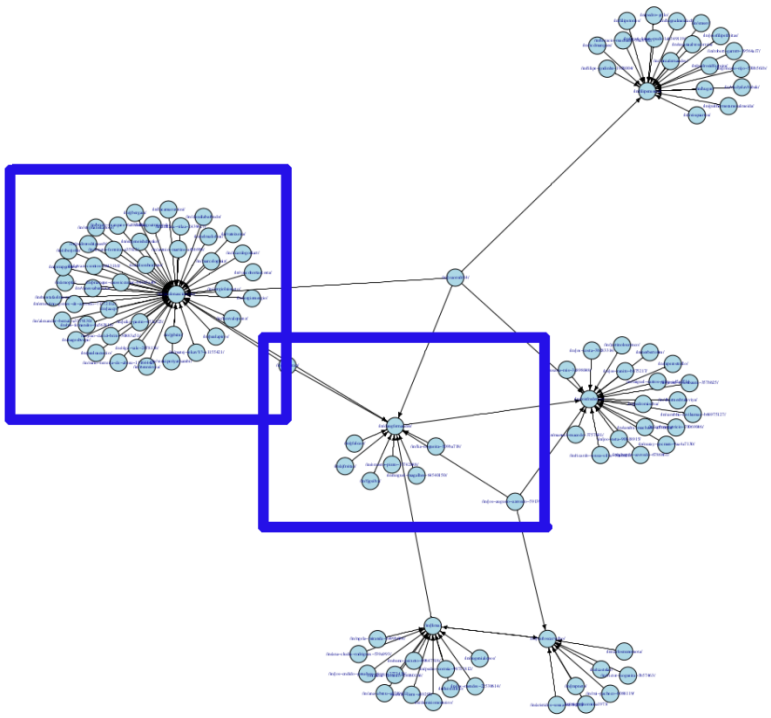

Figura 5.1 - Grafo da especialidade SQL

Tabela 5.2 - Candidatos para SQL com melhor PR

Membro	PR
/in/antonio/	0.188669791419015
/in/david/	0.159361734145306
/in/jose/	0.118292130037489
/in/lima/	0.0860473423756199
/in/carvalho/	0.0594866630545644
/in/isaura/	0.0523146648073396

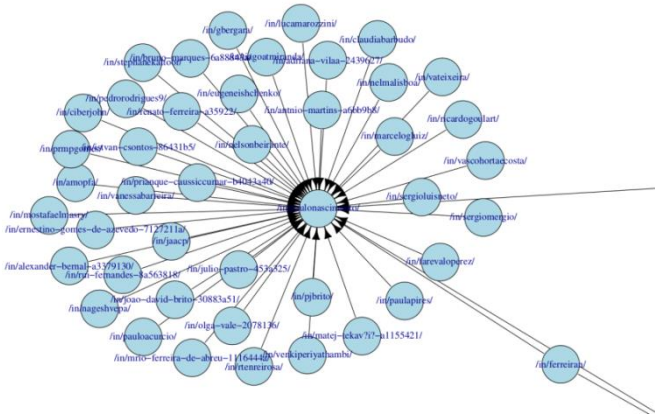

Figura 5.2 - Detalhe A (/in/jose/)

Na Figura 5.2 acima representa-se um detalhe da rede da especialidade SQL centrada num vértice com grau elevado, 45, enquanto abaixo, na Figura 5.3, o detalhe se centra num vértice com menor grau, 10.

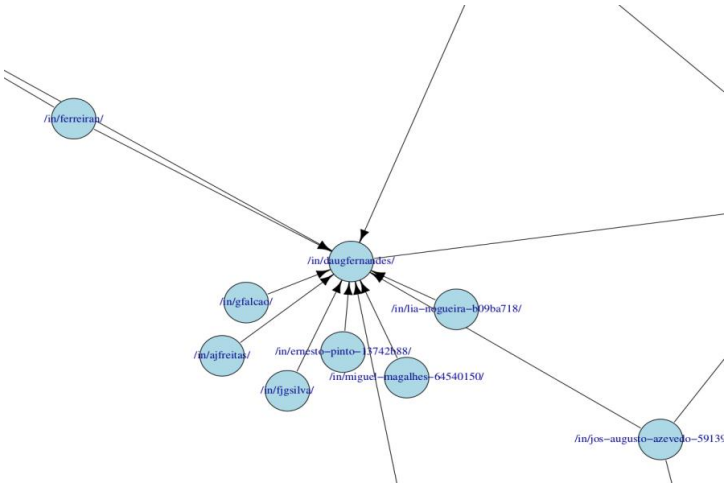

Figura 5.3 - Detalhe B (/in/david/)

Ainda que os cálculos se tenham baseado num conjunto limitado de dados, estas imagens permitem ilustrar uma muito importante característica do algoritmo PageRank que é a de não considerar apenas o número de recomendações mas também a qualidade de cada uma delas.

De facto, se o fizesse, o profissional recomendado na Figura 5.2, com 45 recomendações, deveria ter um valor de PageRank superior ao da Figura 5.3, com apenas 10, o que não se verifica, tal como se pode constatar na Tabela 5.2.

Note-se que um processo de avaliação deste tipo que considere apenas o número de recomendações estará sempre dependente da veracidade das mesmas e em grande medida sujeito a falsas recomendações. É sobre este problema que o algoritmo PageRank vem atuar como vimos acima com um exemplo prático em que o elemento da rede com maior número de recomendações não é o mais valorizado.

5.1.2 Exemplo de recomendações da competência Java

Nesta visualização mostra-se a rede de recomendações da especiadade JAVA, na qual se destacam 3 vértices principais, todos eles com um número semelhante de recomendações.

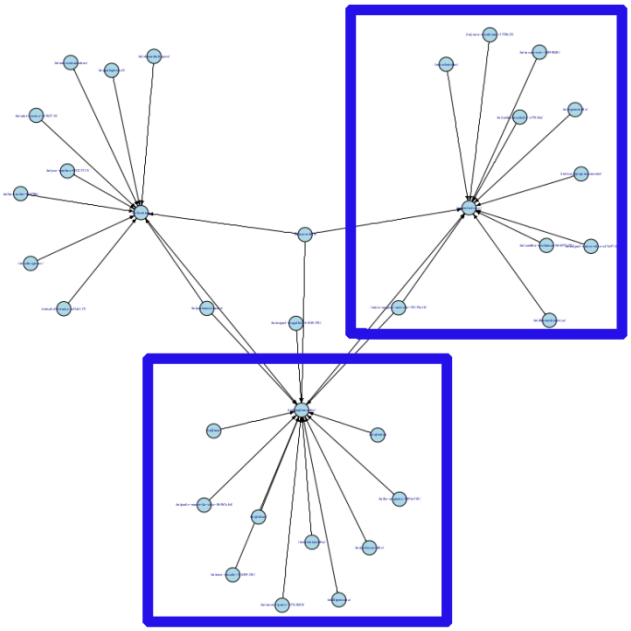

Figura 5.4 - Grafo da especialidade Java

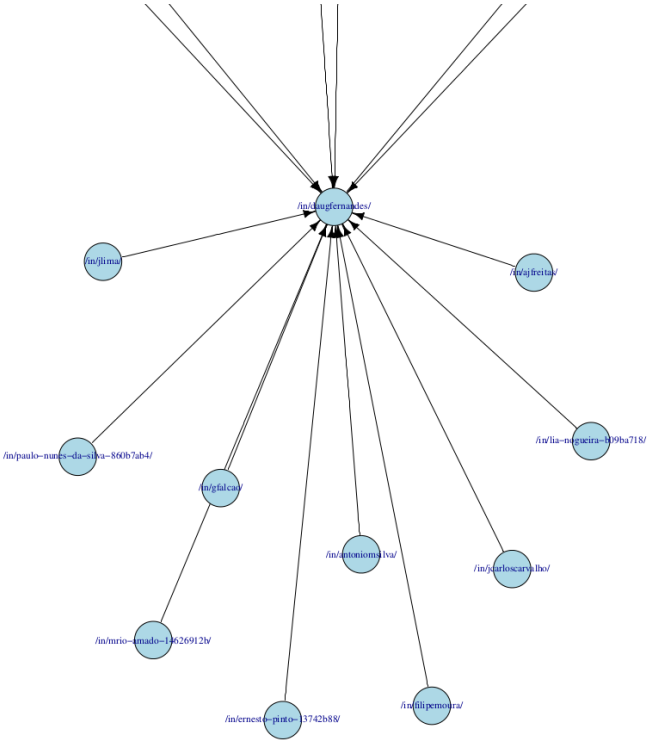

Figura 5.5 - Detalhe C (/in/david/)

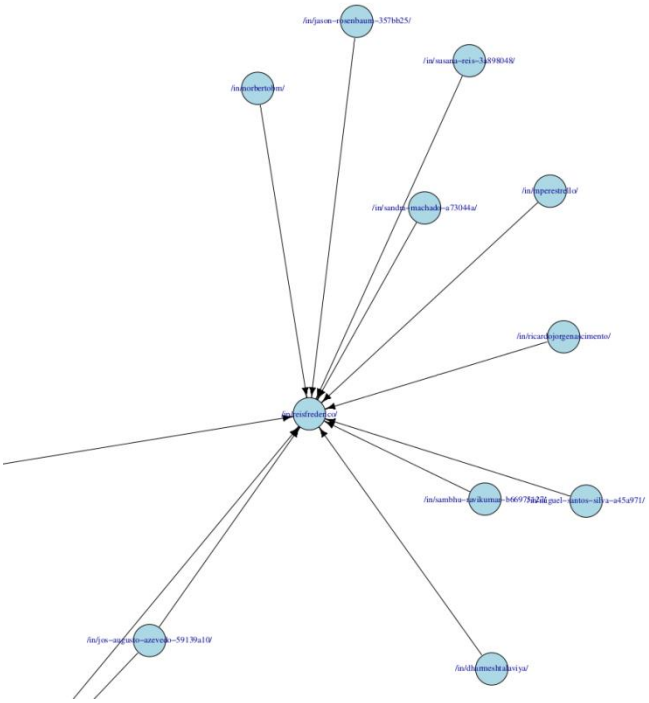

Figura 5.6 - Detalhe D (/in/antonio/)

Também neste caso se consegue perceber a independência do valor de PageRank em relação à quantidade de recomendações.

Tabela 5.3 - Candidatos para Java com melhor *page-rank*

Membro	PageRank
/in/antonio/	0.302627629145042
/in/david/	0.224163069063256
/in/ferreira/	0.101908183321942
/in/mario/	0.0119774554345084
/in/falcao/	0.0119774554345084
/in/ernesto/	0.0119774554345084

5.2 Redes com 2 ou mais atributos (MCDA)

Nesta secção vamos expandir a análise às quatro especialidades escolhidas: SQL, Java, MySQL e C#, e através da utilização do algoritmo TOPSIS procurar os melhores candidatos para algumas combinações de especialidades.

5.2.1 Critérios negativos

O algoritmo TOPSIS permite que se considerem valorizações dos critérios positivas ou negativas. De facto, não tanto por excesso de competências, mas antes pelo que o conhecimento em determinada área pode indiciar de menos positivo para uma determinada necessidade, esta possibilidade de atribuir importâncias negativas a determinadas competências pode, em alguns casos, ser de grande utilidade nomeadamente em face de candidatos com classificações muito semelhantes.

Um caso concreto é o de competência em “Gestão de Projectos”, algo que pode ser visto como negativo para uma posição de “Programador” a inserir numa equipe que se queira homogénea.

5.2.2 Dados utilizados

Os dados utilizados para esta análise consistem nos 20 melhores valores de PageRank para cada uma das quatro especialidades: Tabela 5.4, Tabela 5.5, Tabela 5.6 e Tabela 5.7.

Tabela 5.4 - 20 melhores candidatos em Java

#	Candidato	PageRank
[1,]	/in/antonio/	0.302627629145042
[2,]	/in/david/	0.224163069063256
[3,]	/in/ferreira/	0.101908183321942
[4,]	/in/mario/	0.0119774554345084
[5,]	/in/falcao/	0.0119774554345084
[6,]	/in/ernesto/	0.0119774554345084
[7,]	/in/jose/	0.0119774554345084
[8,]	/in/isaura/	0.0119774554345084
[9,]	/in/lima/	0.0119774554345084
[10,]	/in/nunes/	0.0119774554345084
[11,]	/in/carvalho/	0.0119774554345084
[12,]	/in/joao/	0.0119774554345084
[13,]	/in/azevedo/	0.0119774554345084
[14,]	/in/freitas/	0.0119774554345084
[15,]	/in/nogueira/	0.0119774554345084
[16,]	/in/silva/	0.0119774554345084
[17,]	/in/magalhaes/	0.0119774554345084
[18,]	/in/paulino/	0.0119774554345084
[19,]	/in/alexandre/	0.0119774554345084
[20,]	/in/martins/	0.0119774554345084

Tabela 5.5 - 20 melhores candidatos em C#

#	Candidato	PageRank
[1,]	/in/antonio/	0.434098051539912
[2,]	/in/david/	0.422878692646135
[3,]	/in/falcao/	0.00348837209302326
[4,]	/in/aguiar/	0.00348837209302326
[5,]	/in/ernesto/	0.00348837209302326
[6,]	/in/joao/	0.00348837209302326
[7,]	/in/marinheiro/	0.00348837209302326
[8,]	/in/isaura/	0.00348837209302326
[9,]	/in/lima/	0.00348837209302326
[10,]	/in/gabriel/	0.00348837209302326
[11,]	/in/modesto/	0.00348837209302326
[12,]	/in/ferreira/	0.00348837209302326
[13,]	/in/luis/	0.00348837209302326
[14,]	/in/freitas/	0.00348837209302326
[15,]	/in/garcez/	0.00348837209302326
[16,]	/in/azevedo/	0.00348837209302326
[17,]	/in/nogueira/	0.00348837209302326

[18,]	/in/patricia/	0.00348837209302326
[19,]	/in/ines/	0.00348837209302326
[20,]	/in/silva/	0.00348837209302326

Tabela 5.6 - 20 melhores candidatos em Sql

#	Candidato	PageRank
[1,]	/in/antonio/	0.188669791419015
[2,]	/in/david/	0.159361734145306
[3,]	/in/jose/	0.118292130037489
[4,]	/in/lima/	0.0860473423756199
[5,]	/in/carvalho/	0.0594866630545644
[6,]	/in/isaura/	0.0523146648073396
[7,]	/in/felix/	0.0031681856052893
[8,]	/in/falcao/	0.0031681856052893
[9,]	/in/ernesto/	0.0031681856052893
[10,]	/in/joao/	0.0031681856052893
[11,]	/in/ferreira/	0.0031681856052893
[12,]	/in/freitas/	0.0031681856052893
[13,]	/in/nogueira/	0.0031681856052893
[14,]	/in/magalhaes/	0.0031681856052893
[15,]	/in/azevedo/	0.0031681856052893
[16,]	/in/ilda/	0.0031681856052893
[17,]	/in/felizberto/	0.0031681856052893
[18,]	/in/angela/	0.0031681856052893
[19,]	/in/diana/	0.0031681856052893
[20,]	/in/manuela/	0.0031681856052893

Tabela 5.7- 20 melhores candidatos em MySql

#	Candidato	PageRank
[1,]	/in/david/	0.408677955063497
[2,]	/in/antonio/	0.357888309996744
[3,]	/in/isaura/	0.0845030120481928
[4,]	/in/carvalho/	0.00615963855421687
[5,]	/in/mario/	0.00180722891566265
[6,]	/in/falcao/	0.00180722891566265
[7,]	/in/ernesto/	0.00180722891566265
[8,]	/in/aguiar/	0.00180722891566265
[9,]	/in/joao/	0.00180722891566265
[10,]	/in/martinho/	0.00180722891566265
[11,]	/in/lima/	0.00180722891566265
[12,]	/in/modesto/	0.00180722891566265
[13,]	/in/santos/	0.00180722891566265
[14,]	/in/freitas/	0.00180722891566265
[15,]	/in/luis/	0.00180722891566265
[16,]	/in/silva/	0.00180722891566265
[17,]	/in/magalhaes/	0.00180722891566265

[18,]	/in/nogueira/	0.00180722891566265
[19,]	/in/azevedo/	0.00180722891566265
[20,]	/in/garcez/	0.00180722891566265

5.2.3 Explorações

As explorações que a seguir se descrevem visam cobrir várias combinações significativas de critérios:

- pesos idênticos;
- pesos diferenciados;
- pesos negativos.

Primeiramente procedeu-se à junção, por **Candidato,** das tabelas: Tabela 5.4, Tabela 5.5, Tabela 5.6 e Tabela 5.7 e, por facilidade, consideraram-se apenas os candidatos que possuem valor de PageRank em todas as especialidades. Não consideramos, portanto, candidatos com valores omissos.

Uma vez que as análises nem sempre incidem sobre todos os atributos, a decisão de não incluir elementos com informação incompleta pode excluir candidatos válidos para análise de determinadas combinações de critérios. Numa implementação real, todos os candidatos devem ser considerados, independentemente do número de especialidades para as quais foram recomendados, devendo as rotinas implementadas ser capazes de levar esse facto em linha de conta.

A tabela resultante:

Tabela 5.8 - Tabela de candidatos com conhecimentos de sql, mysql, csharp e java

Candidato	sql	mysql	csharp	java
/in/freitas/	0.003168186	0.001807229	0.003488372	0.01197746
/in/david/	0.159361734	0.408677955	0.422878693	0.22416307
/in/ernesto/	0.003168186	0.001807229	0.003488372	0.01197746
/in/isaura/	0.052314665	0.084503012	0.003488372	0.01197746
/in/falcao/	0.003168186	0.001807229	0.003488372	0.01197746
/in/lima/	0.086047342	0.001807229	0.003488372	0.01197746
/in/joao/	0.003168186	0.001807229	0.003488372	0.01197746
/in/azevedo/	0.003168186	0.001807229	0.003488372	0.01197746

/in/nogueira/	0.003168186	0.001807229	0.003488372	0.01197746
/in/antonio/	0.188669791	0.357888310	0.434098052	0.30262763

Todos os critérios com pesos idênticos

Nesta análise procura analisar-se uma necessidade em que todos os critérios são considerados como positivos, isto é, representam características desejadas para o perfil pretendido.

Exemplo: escolher os 4 melhores candidatos com conhecimentos em Sql e Java.

```
1: # todos critérios com igual peso
2: weights <- c(0.25,0.25,0.25,0.25)
3:
4: names(weights) <- colnames(madf)
5: criteriaMinMax <- c("max","max","max","max")
6: names(criteriaMinMax) <- colnames(madf)
7:
8: # utilizar apenas os critérios sql e java
9: criteriaIDs <- c("sql","java")
10:
11: result <-
12:   sort(
13:         TOPSIS(performanceTable =  madf,
14:               criteriaWeights = weights,
15:               criteriaMinMax = criteriaMinMax,
16:               criteriaIDs = criteriaIDs),
17:         decreasing = TRUE)
18:   [seq(4)]
```

Código Fonte 5.1 - 4 melhores candidatos Sql e Java

Obtendo os seguintes resultados:

```
                    result
/in/antonio/ 1.0000000
/in/david/ 0.7754720
/in/lima/         0.2654553
/in/isaura/   0.1663549
```

Na linha 2: define-se o vector de pesos, neste caso idênticos para as quatro especialidades e na linha 5: o vector de critérios, no caso todos “max”, o que significa que o algoritmo escolherá ps elementos que maximizam cada um dos atributos.

Nas linhas 4: e 6: apenas se nomeiam os elementos dos vectores concomitantemente com as colunas da dataframe “madf” que possui os dados (ver Tabela 5.8).

Na linha 9: define-se o vector que indica que atributos pretendemos utilizar e nas linhas 12: a 16: invoca-se a função que executa a análise TOPSIS com os argumentos preparados anteriormente.

O vector resultado é ordenado (linha 12:) por ordem decrescente (linha 17:) do qual se extraem os primeitos 4 elementos (linha 18:).

Critérios mais importantes com pesos diferenciados

Nesta outra situação, em que também se consideram todos os critérios como positivos, mostra-se a possibilidade de diferenciar a importância que se pretende atribuir a cada um deles. No caso, pretende-se um candidato com conhecimentos de Java, o critério mais importante, mas também com conhecimentos de MySql embora tal não seja uma característica tão valorizada (um *plus*, digamos assim).

Exemplo: Escolher os 4 melhores candidatos com conhecimentos em Java, considerando-se conhecimentos em MySql opcionais.

```
1: # critérios com pesos diferenciados
2: weights <- c(0.25,0.05,0.25,0.45)
3:
4: names(weights) <- colnames(madf)
5: criteriaMinMax <- c("max","max","max","max")
6: names(criteriaMinMax) <- colnames(madf)
7:
8: # utilizar apenas os critérios mysql e java
9: criteriaIDs <- c("mysql","java")
10:
11: result <-
12:   sort(
13:         TOPSIS(performanceTable =  madf,
14:               criteriaWeights = weights,
15:               criteriaMinMax = criteriaMinMax,
16:               criteriaIDs = criteriaIDs),
17:         decreasing = TRUE)
18:   [seq(4)]
```

Código Fonte 5.2 - 4 melhores candidatos Java e opcionalmente com conhecimentos de MySql

Obtendo os seguintes resultados:

```
                    result
/in/antonio/ 0.9868788
/in/david/ 0.7321262
/in/isaura/   0.0212048
/in/freitas/     0.0000000
```

Este exemplo apenas difere do anterior nos pesos atribuidos a cada um dos atributos. Na linha 2: atribuem-se um menor peso ao atributo mySql enquanto se incrementa o atributo Java.

Critérios considerados como características im desejáveis, negativas

Neste caso, mostra-se como proceder no caso de se pretender considerar determinado critério como negativo. Concretamente, considera-se que possuir conhecimentos de MySql seria prejudicial para o perfil pretendido.

Exemplo: Escolher os 4 melhores candidatos com conhecimentos em C# e Sql mas não MySQL.

```
1: # todos critérios com igual peso
2: weights <- c(0.25,0.25,0.25,0.25);
3:
4: names(weights) <- colnames(madf)
5: criteriaMinMax <- c("max","min","max","max")
6: names(criteriaMinMax) <- colnames(madf)
7:
8: # utilizar os critérios csharp, sql e mysql
9: criteriaIDs <- c("csharp","sql", "mysql")
10:
11: result <-
12:   sort(
13:         TOPSIS(performanceTable =  madf,
14:               criteriaWeights = weights,
15:               criteriaMinMax = criteriaMinMax,
16:               criteriaIDs = criteriaIDs),
17:         decreasing = TRUE)
18:   [seq(4)]
```

Código Fonte 5.3 - 4 melhores candidatos C# e Sql mas não MySql

Obtendo os seguintes resultados:

```
                    result
/in/antonio/ 0.6065130
/in/david/ 0.5477287
/in/lima/          0.4983597
/in/freitas/      0.4267702
```

Neste caso, a minimização de um atributo faz-se na linha 5: através da indicação "min" no vector de critérios *minmax.*

A título de exemplo, e com base no exemplo acima, vejamos como poderíamos aumentar a negatividade do conhecimento em MySql apenas alterando os pesos para **`weights <- c(0.20,0.40,0.20,0.20)`** e confira-se como o resultado se altera drasticamente, passando a ser considerados os candidatos com menor valor de PageRank na especialidade MySql, como se pretendia.

```
                              result
/in/lima/                   0.6518101
/in/freitas/               0.5982326
/in/ernesto/ 0.5982326
/in/falcao/                 0.5982326
```

Capítulo 6

Conclusões e trabalhos futuros

6 Conclusões e trabalhos futuros

Este trabalho mostra a possibilidade de aplicação de técnicas de análise objetiva de redes a um problema normalmente analisado sob uma perspetiva subjetiva: a avaliação de candidatos em processos de seleção de recursos humanos baseada nas recomendações da rede social LinkedIn.

Através dos métodos propostos, pretendeu-se não apenas acelerar o processo de identificação dos melhores candidatos para um determinado perfil mas também a libertação da componente subjetiva dos métodos tradicionais. Naturalmente que não fica eliminada a necessidade de uma análise posterior mais detalhada dos candidatos, mas certamente ajuda a reduzir o número de potenciais candidatos a avaliar sob outros prismas.

Para tal, aplicou-se com sucesso o algoritmo PageRank à análise de uma rede de recomendações profissionais entre pares bem como o algoritmo TOPSIS na decisão multi-atributo para duas ou mais características necessárias a um determinado perfil profissional.

Os dados sobre os quais incidiram as análises foram obtidos de algumas páginas de membros do LinkedIn e esta extração de dados foi efetuada através de *data-mining* HTML aplicando-se algumas técnicas com interesse.

Visto tratar-se de um protótipo, a construção das rotinas implementadas não teve preocupações de usabilidade e pretendeu apenas dotá-las de um mínimo de comodidade de utilização. Uma implementação virada para a utilização em ambiente real necessitará obrigatoriamente dessa componente o que passará pelo desenvolvimento de uma aplicação completa com o respetivo interface de utilização.

6.1 Trabalhos futuros

A estrutura de armazenamento dos dados é simples mas eficaz para os objetivos pretendidos. Numa aplicação em larga escala o suporte teria de ser obrigatoriamente outro que não um simples ficheiro JSON. Uma base de dados documental NoSql como a

MongoDB (2017) ou CouchDB (Apache, 2017) parece-nos uma boa solução, ambos os casos com conectores para R.

Por outro lado, a utilização de bases de dados será obrigatória na implementação de qualquer solução completa que contemple também a componente de manutenção do repositório de recomendações, algo que não se encontra no âmbito do trabalho aqui apresentado.

As avaliações produzidas pelos métodos propostos são tanto mais fiáveis quanto mais extensa for a rede de recomendações. Os métodos propostos carecem de testes com a utilização de volumes de dados de maior dimensão.

Com vista a enriquecer a rede, procurar tirar partido de eventuais relações entre os grafos das várias especialidades. A ideia de que um bom recomendador Hibernate será certamente um bom recomandador JAVA ou um recomendador de C# será também de .NET Framework, parece indiciar uma forma de colmatar alguns problemas levantados pela utilização de redes de especialidades com reduzida dimensão. Trabalhos nesta área indicam isso mesmo: Pérez-Rosés et al. (2016) e Pérez-Rosés e Sebé (2017).

A simplicidade do modelo de dados e a baixa complexidade das rotinas implementadas permite a muito rápida aplicação destes métodos a um processo de RH tradicional já existente e assim proceder a uma avaliação comparada de resultados.

Bibliografia

Ankolekar, A., e Vrandecic, D. (2008). Kalpana - enabling client-side web personalization. In *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia - HT '08* (p. 21). New York, New York, USA: ACM Press. https://doi.org/10.1145/1379092.1379100

Apache (2017). Apache CoucheDB. http://couchdb.apache.org/

Bakar, A. A. and Ting, C.-Y. (2011). Soft skills recommendation systems for it jobs: A bayesian network approach. *In 2011 3rd Conference on Data Mining and Optimization (DMO)*, pag 82–87.

Ban, C. (2006). Hiring in the Federal Government: Political and technological sources of reform. *Public personnel management: Current challenges, future challenges.*

Barnes, N. G. e Mattson, E. (2009). Social Media Adoption Among the 2009 Inc. 500: New Tools and New Trends. www.sncr.org/sites/default/files/socialmedia2009_0.pdf

Bastian, M., Hayes, M., Vaughan, W., Shah, S., Skomoroch, P., Kim, H., Uryasev, S., and Lloyd, C. (2014). LinkedIn skills: Large-scale topic extraction and inference. *8th ACM Conference on Recommender Systems.*

Belton, V., e Stewart, T. J. (2002). Multiple Criteria Decision Analysis. Boston, MA: Springer US. https://doi.org/10.1007/978-1-4615-1495-4

Bersin, J. (2012). LinkedIn is Disrupting the Corporate Recruiting Market. http://www.forbes.com/sites/joshbersin/2012/02/12/linkedin-is-disrupting-the-corporate-recruiting-market.

Biggs, N., Lloyd, E. K., e Wilson, R. J. (1986). *Graph Theory, 1736-1936*. New York, NY, USA: Clarendon Press.

Bollobás, B. (1998). *Modern Graph Theory* (Vol. 184). New York, NY: Springer New York. https://doi.org/10.1007/978-1-4612-0619-4

Bose, N. K., e Liang, P. (1996). Neural Network Fundamentals with Graphs, Algorithms, and Applications. Hightstown, NJ, USA: McGraw-Hill, Inc.

Brin, S. e Page, L. (1998). The Anatomy of a Large-scale Hypertextual Web Search Engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117.

Capiluppi, A., Serebrenik, A., e Singer, L. (2013). Assessing Technical Candidates on the Social Web. *IEEE Software*, 30(1):45–51.

Carley, K. M., e Prietula, M. J. (1994). Computational organization theory. *Computational organization theory*. L. Erlbaum Associates.

Casteleyn, S., Daniel, F., Dolog, P., e Matera, M. (2009). Engineering Web Applications (1st ed.). *Springer Publishing Company*, Incorporated.

Comunidade. (2005). Greasespot. Consultado em 8 de fevereiro de 2017 em http://www.greasespot.net/

Csardi, G., e Nepusz, T. (2006). The igraph software package for complex network research. InterJournal, Complex Sy, 1695.

Domeniconi, G., Moro, G., Pagliarani, A., Pasini, K., and Pasolini, R. (2016). Job recommendation from semantic similarity of linkedin users' skills. In *Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods* - Volume 1: ICPRAM,, pages 270–277. INSTICC, SciTePress.

Dumais, S. T. (2005). *Latent semantic analysis*. Annual Review of Information Science and Technology, 38(1):188–230.

Easley, D., e Kleinberg, J. (2010). Networks, Crowds, and Markets: Reasoning About a Highly Connected World. New York, NY, USA: Cambridge University Press.

Farquhar, P. H. (1977). A survey of multiattribute utility theory and applications. In M. K. Starr e M.Zeleny (Eds.), *TIMS Studies in the management sciences (Vol. 6, pp. 59-90).* North-Holland Publishing Company.

Fernandes, D. (2015), Pagerank na seleção de recursos humanos, *InforAberta 2015 – Jornadas de Informática da Universidade Aberta (Montijo),*

https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxpbmZvcmFiZXJ0YTIwMTV8Z3g6NzhkODI1MGRkM2E1MDVmNQ

Firmenich, D., Firmenich, S., Rivero, J. M., e Antonelli, L. (2014). A Platform for Web Augmentation Requirements Specification (pp. 1–20). Springer, Cham. https://doi.org/10.1007/978-3-319-08245-5_1

Gkorou, D., Pouwelse, J., e Epema, D. (2015). Trust-based Collection of Information in Distributed Reputation Networks. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, pages 2312–2319, New York, NY, USA. ACM.

Gusdorf, M., Schaefer, B., Woolever, N. A., e Leonard, S. H. (2008). *Recruitment and Selection: Hiring the Right Person A two-part learning module for undergraduate students*. Society for Human Resource Management.

Hwang, C.-L. and Yoon, K. (1981). Methods for Multiple Attribute Decision Making. pages 58–191.

Karaa, W. B. A. e Amdouni, S. (2011). E-recruiting support system based on text mining methods. *IJKL*, 7(3/4):220–232.

Kessler, R., Béchet, N., Torres-Moreno, J. M., Roche, M., e El-Bèze, M. (2009). Profilage de candidatures assisté par Relevance Feedback. *TALN 2009*, Senlis.

Köksalan, M., Wallenius, J., and Zionts, S. (2013). An Early History of Multiple Criteria Decision Making. *Journal of Multi-Criteria Decision Analysis*, 20(1-2):87–94.

Kubica, J., Moore, A., Cohn, D., e Schneider, J. (2003). Finding Underlying Connections: A Fast Graph-Based Method for Link Analysis and Collaboration Queries. In *Proceedings of the International Conference on Machine Learning.*

Li, L., Shang, Y., e Zhang, W. (2002). Improvement of HITS-based algorithms on web documents. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 527–535, New York, NY, USA. ACM.

LinkedIn (2016). LinkedIn Developers. Consultado em 27 de Junho de 2016 em https://developer.linkedin.com/.

Meyer, P., Bigaret, S., Hodgett, R., A.-L. O. (2017). CRAN - Package MCDA. Consultado em 27 de Junho de 2016 em https://cran.r-project.org/web/packages/MCDA/index.html

MongoDB (2017). MongoDB. https://www.mongodb.com/

Network, M. D. (2016). HTTP access control (CORS) - HTTP | MDN. Consultado em 9 de Dezembro de 2016 em https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS

Newman, M. E. J. (2003). The structure and function of complex networks. SIAM REVIEW, 45, 167–256.

Oliveira, E. (2010). Uma Visão sobre Teorias acerca do desenvolvimento e da caracterização da Investigação Científica. Technical report, Faculdade de Engenharia da Universidade do Porto.

Oracle (2013). Modern HR in the Cloud: Best Practices for Recruiting the Best Talent. Technical report, Oracle Corporation.

Otsuka, T., Yoshimura, T., e Ito, T. (2012). Evaluation of the Reputation Network Using Realistic Distance Between Facebook Data. In *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Volume 03*, WI-IAT '12, pages 383–388, Washington, DC, USA. IEEE Computer Society.

Pérez-Rosés, H., Sebé, F., and Ribó, J. M. (2016). Endorsement deduction and ranking in social networks. *Computer Communications*, 73:200 – 210. Online Social Networks.

Pérez-Rosés, H. and Sebé, F. (2017). Iterated endorsement deduction and ranking. Computing, 99(5):431–446.

Samatova, N. F., Hendrix, W., Jenkins, J., Padmanabhan, K., e Chakraborty, A., editors (2014). *Practical Graph Mining with R*, chapter Link analy, pages 75–131. CRC Press.

Strohmeier, S. e Piazza, F. (2013). Domain driven data mining in human resource management: A review of current research. *Expert Systems with Applications*, 40(7):2410–2420.

Tsugawa, J. e III, H. C. R. (2004). Identifying Talent through Technology: Automated Hiring Systems in Federal Agencies. Technical report, U.S. Merit Systems Protection Board, Washington, DC (1615 M. St., N.W., Washington 20419).

W3C. (1998). Level 1 Document Object Model Specification. Consultado em 1 de Julho de 2016 em https://www.w3.org/TR/WD-DOM/

Zide, J., Elman, B., and Shahani-Denning, C. (2014). Linkedin and recruitment: how profiles differ across occupations. *Employee Relations*, 36(5):583–604.

Anexos

A – Código fonte

userscript.js

```
/* LinkedIn recomendations crawler
 * version 0.1
 * 2015-10-14
 * David Fernandes <902006@estudante.uab.pt>
 *
 * --------------------------------------------------------------------
 *
 * This is a Greasemonkey user script.
 *
 * --------------------------------------------------------------------
 */
// ==/UserScript==
// @name          LinkedIn recomendations crawler
// @namespace     https://www.linkedin.com
// @description   Get recommendations from LinkedIn user page
// @include       https://www.linkedin.com/*
// @exclude       *
// ==/UserScript==

var ajaxurl = "https://localhost/linkedin.php";

function send(origem, endorsed, skill, data) {
    callAjax(ajaxurl, origem, endorsed, skill, data, function (x) {
    });
}

var endorsedAndSkills = [];

var areasCovered = [];
var endorsersCovered = [];
var totalEndorsersCovered = {};
var areaOfExpertise = "err";
var numberOfEndorsers = 0;
var endorsersHandled = 0;
var endorsers = [];

function eventFire(el, etype) {
    if (el.fireEvent) {
        el.fireEvent('on' + etype);
    } else {
        var evObj = document.createEvent('Events');
        evObj.initEvent(etype, true, false);
        el.dispatchEvent(evObj);
        console.log("Skills dialog expanded!");
    }
}

unsafeWindow.countRecommendations = function () {

    console.log("Waiting for dialog.");

    var dialog_recommendations = document.getElementsByClassName("pv-profile-
detail__modal")[0]

    if (dialog_recommendations) {
        var profile_detail_header = dialog_recommendations.getElementsByTagName("h2")[0];
        if (profile_detail_header) {
            var span_expertise = profile_detail_header.getElementsByTagName("span")[0];
            if (span_expertise) {
                areaOfExpertise = span_expertise.innerHTML;
                var myRegexp = /\s*(.*)\s\((.*)\)/g;
                var match = myRegexp.exec(areaOfExpertise);
                areaOfExpertise = match[1];
                numberOfEndorsers = match[2];
            }
```

```
        }
    }

    var skillsPanel = document.getElementsByClassName("pv-featured-skills-section")[0];
    if (skillsPanel) {
        var elementSeeMoreA = skillsPanel.querySelectorAll("button[aria-controls='featured-
skills-expanded']")[0];
        if (elementSeeMoreA) {
            var spans = elementSeeMoreA.getElementsByTagName("span");
            var idx;
            for (idx = 0; idx < spans.length; idx++) {
                span = spans[idx];
                if (span) {
                    var text = span.innerHTML;
                    if (text) {
                        if (text.indexOf("more") > -1) {
                            eventFire(elementSeeMoreA, 'click');
                            break;
                        }
                    }
                }
            }
        }
    }

    if (!totalEndorsersCovered[areaOfExpertise]) {
        totalEndorsersCovered[areaOfExpertise] = 0;
    }

    if (numberOfEndorsers > totalEndorsersCovered[areaOfExpertise]) {
        var endorsersDiv = document.getElementsByClassName("pv-profile-
detail__content")[0];
        if (endorsersDiv) {
            endorsersDiv.scrollTop = endorsersDiv.scrollHeight;
        }
    }

    if (areaOfExpertise !== "err" && (!contains(areasCovered, areaOfExpertise) ||
numberOfEndorsers > totalEndorsersCovered[areaOfExpertise])) {

        if (!contains(areasCovered, areaOfExpertise)) {
            endorsersCovered = [];
            endorsersHandled = 0;
            areasCovered.push(areaOfExpertise);
            totalEndorsersCovered[areaOfExpertise] = 0;
            endorsers = [];
            console.log("Area of expertise => ", areaOfExpertise);
        }

        var endorsersLI = document.getElementsByClassName("pv-endorsement-entity");
        //console.log("endorsersLI", endorsersLI);

        if (endorsersLI.length > 0) {
            var idx;
            for (idx = 0; idx < endorsersLI.length; idx++) {

                var endorser = endorsersLI[idx].getElementsByTagName("a")[0];
                if (endorser) {

                    var go = false;

                    var endorserID = endorser.getAttribute("href");
                    var endorserName = "";
                    var endorserHeadline = "";

                    if (endorserID) {
                        if (!contains(endorsersCovered, endorserID)) {
                            endorsersHandled++;
                            totalEndorsersCovered[areaOfExpertise] = endorsersHandled;
                            console.log(endorserID);
                            endorsersCovered.push(endorserID);
                            go = true;
                        }
                    } else
```

```
                        console.log("no endorserID <a>");

                    if (go) {

                        var endorserDivName = endorser.getElementsByClassName("pv-
endorsement-entity__name")[0];
                        if (endorserDivName) {
                            var endorserNameSpan =
endorserDivName.getElementsByTagName("span")[0];
                            if (endorserNameSpan) {
                                endorserName = endorserNameSpan.innerHTML;
                            }
                        }

                        var endorserDivHeadline = endorser.getElementsByClassName("pv-
endorsement-entity__headline")[0];
                        if (endorserDivHeadline) {
                            endorserHeadline = endorserDivHeadline.innerHTML;
                        }
                        endorsers.push('{"id":"' + endorserID + '","name":"' + endorserName
+ '"}');

                    }
                }
            }
        }
    }


    if (numberOfEndorsers !== 0 && numberOfEndorsers -
totalEndorsersCovered[areaOfExpertise] === 0) {
        var me = "/in/" + window.location.href.split("/in/")[1].split("/")[0] + "/";

        if (!contains(endorsedAndSkills, me + "|" + areaOfExpertise)) {
            endorsedAndSkills.push(me + "|" + areaOfExpertise);
            console.log("{\"id\":\"" + me + "\",\"area\":\"" + areaOfExpertise + "\",
\"endorsers\":[" + endorsers + "]}");
            send('ubuntu_david', me, areaOfExpertise, '{"id":"' + me + '","area":"' +
areaOfExpertise + '", "endorsers":[' + endorsers + ']}');
            numberOfEndorsers = 0;
        }

        numberOfEndorsers = 0;
        endorsers = [];

    }

    setTimeout(function () {
        unsafeWindow.countRecommendations();
    }, 5000);

}

if (window.location.href.indexOf("linkedin") > -1) {

    var xmlhttp2;
    xmlhttp2 = new XMLHttpRequest();
    xmlhttp2.onreadystatechange = function () {
        if (xmlhttp2.readyState == 4 && xmlhttp2.status == 200) {
            endorsedAndSkills = xmlhttp2.responseText.split(";");
            xmlhttp2 = null;
            setTimeout(function () {
                unsafeWindow.countRecommendations();
            }, 5000);
        }
    }
    xmlhttp2.open("GET", "https://localhost/linkedin.php", true);
    xmlhttp2.send();
}

function callAjax(url, origem, endorsed, skill, data, callback) {

    var xmlhttp;
    xmlhttp = new XMLHttpRequest();
```

```
    xmlhttp.onreadystatechange = function () {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
            callback(xmlhttp.responseText);
        }
    }
    xmlhttp.open("POST", url, true);
    xmlhttp.send(JSON.stringify({"zaxs": "1", "endorsed": endorsed, "skill": skill,
"origem": origem, "data": data}));
}

function contains(a, obj) {
    for (var i = 0; i < a.length; i++) {
        if (a[i] === obj) {
            return true;
        }
    }
    return false;
}
```

linkedin.php

```
<?php

header("Content-Type: application/json; charset=utf-8", true);
header('Access-Control-Allow-Origin: *', true);

$servername = "localhost";
$username = "user1";
$password = "xxxxxxxx";
$dbname = "mestrado";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
$conn->set_charset("utf8");

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

switch ($_SERVER['REQUEST_METHOD']) {

    case "POST":
        $post_vars = json_decode(file_get_contents("php://input"));

       $a = (object)$post_vars;

        if ($a->zaxs==1) {
           $endorsed = urldecode($a->endorsed);
           $skill = urldecode($a->skill);
           $origem = urldecode($a->origem);
           $data = urldecode($a->data);
           echo insert($conn, $endorsed, $skill, $origem, $data);
       } else
         die("\n\nWrong arguments.");

       case "GET":
           echo get($conn);

}

$conn->close();

function get($mysqli) {
       try {
              $res='';
              $sql = 'SELECT DISTINCT endorsed, skill, data from linkedin order by
endorsed, skill';
              $result = mysqli_query($mysqli, $sql);

              if (mysqli_num_rows($result) > 0)
```

```
                {
                        while($row = mysqli_fetch_assoc($result))
                        {
                                if($res!="") $res.=";";
                                $res  .=  $row['endorsed']  .  '|'  .  $row['skill']  .  '|'  .
$row['data']. '||';
                        }
                        return $res;
                } else {
                return "";
                }
        } catch (Exception $e) {
                return 'Caught exception: '; // . $e->getMessage() . "\n";
        }
}


function insert($mysqli, $endorsed, $skill, $origem, $data) {

    try {

        if  ($stmt  =  $mysqli->prepare("INSERT  INTO  linkedin  (endorsed,skill,origin,data)
VALUES ('" . $endorsed . "','" . $skill . "','" . $origem . "','" . $data . "')")) {
                $stmt->execute();
            $newid2 = $mysqli->insert_id;
            return $newid2;
        } else {
            return 'Error prepare.'; // . $e->getMessage() . "\n";
        }
    } catch (Exception $e) {
        return 'Caught exception: '; // . $e->getMessage() . "\n";
    }
}

?>
```