Banco de dados orientado a grafos com dados de artigos científicos

Evelym Maria de Lourdes Rondon Pereira[1], **Thiago Meirelles Ventura**[1]

[1] Instituto de Computação – Universidade Federal de Mato Grosso (UFMT)
Campus Cuiabá – MT – Brazil

evionline@gmail.com, thiago@ic.ufmt.br

Abstract. *Is there a tool for similarity between articles? In this paper we intend to contribute to this discussion by constructing a non-relational graph-oriented database mechanism that seeks to make links or relationships between articles, as well as which author could evaluate a particular article or which authors have a conflict of interest with others.The great difficulty of the project lies in the part of data processing, which has information taken from scientific journals. This information does not have native relations between them, so it is difficult to collect information that has links between them as common guiding principles to several articles, articles of the same author evaluated by several scientific journals, among other situations. Such information is required for advisor or for the submission of academic articles. Thinking about how to solve the problem of data interconnection should be selected tools that organize and streamline the processing of a large amount of data, so as to be able to correlate the existing links and reach the expected result.*

Keywords*: scientific article, Not Only SQL, Neo4j.*

Resumo. *Há uma ferramenta para similaridade entre artigos? Neste trabalho pretendemos contribuir para essa discussão ao construir um mecanismo via banco de dados não relacional orientado a grafos, que busca realizar relação entre artigos, assim como qual autor poderia avaliar um determinado artigo ou quais autores possuem conflito de interesse com outros. A grande dificuldade do projeto está na parte de processamento de dados, que tem informações retiradas de revistas científicas. Estas informações não possuem relações nativas entre elas, por isso se torna trabalhoso coletar informações que tenham vínculos entre si como autores comuns a vários artigos, artigos de um mesmo autor avaliadas por várias revistas científicas, entre outras situações. Tais informações são necessárias para cientistas ou para a banca de submissão de artigos acadêmicos. Pensando em como resolver o problema de interligação dos dados, foi proposto neste trabalho um banco de dados orientado a grafos para conseguir correlacionar os vínculos existentes. Um estudo de caso foi realizado para validar os resultados.*

Palavras-chave*: artigo científico, Banco de Dados Não Convencionais, Neo4j.*

1. Introdução

A criação de um artigo científico vem para propor novas ideias e solucionar dificuldades encontradas no cotidiano, sob perspectiva de uma metodologia elaborada pelo autor. Para que o artigo seja publicado, há todo um processo de avaliação e revisão com a ação de outros pesquisadores. Dificuldades podem surgir neste processo, como a definição de um melhor avaliador para determinado tema ou a verificação de conflito de interesses. Esses cenários ocorrem pois a diversidade de artigos escritos é imensa e não há, muitas das vezes, um ambiente propício para encontrar determinadas relações.

Por outro lado, um pesquisador ao elaborar um artigo científico, deve possuir o maior domínio possível sobre o assunto a ser apresentado e discutido, e este assim o faz, pesquisando trabalhos, autores e temas correlatos ao seu. Todavia, mediante ao imenso volume de artigos produzidos, este domínio completo fica humanamente inviável, não possibilitando o cruzamento de dados para uma melhor análise [Araujo et al. 2015].

Os artigos possuem conteúdo suficiente para que relações sejam encontradas entre trabalhos e autores. Como exemplo, pode ser citado um artigo que tenha uma determinada palavra-chave que está sendo mencionada em outros artigos. Isso pode significar que os autores de ambos os artigos já trabalharam ou poderão trabalhar juntos. As palavras chaves é uma classificação generalista do que cada artigo aborda como tema, então quanto maior for o número de palavras-chaves relacionada entre os artigos, melhor para determinar suas semelhanças [Mena-Chalco et al. 2012].

Visando facilitar esse processo de avaliação, serão empregadas técnicas de relacionamento de dados utilizando o conceito de grafos, que irá formar padrões, ou seja, relacionar os dados dos artigos científicos. A finalidade deste trabalho se concentra em posicionar características correlatas e relacionar dados de trabalhos científicos por meio de seus atributos em comum, facilitando a pesquisa de artigos que tenham características semelhantes.

Portanto, este trabalho consiste em utilizar dados de artigos acadêmicos em um banco de dados não relacional orientado a grafos, com o propósito de relacioná-los automaticamente. Com isso, é possível obter relações facilmente, como artigos com temas semelhante, autores correspondentes e gerindo até mesmo conflitos de interesses entre autores e revisores.

Este trabalho está organizado da seguinte maneira: embasamento teórico está descrito na seção 2 com a finalidade de apresentar os meios com que o trabalho foi desenvolvido e as ferramentas utilizadas; na seção 3 são abordados os métodos e a solução para o caso de estudo; na seção 4 são apresentados os resultados obtidos; e na seção 5 as considerações finais.

2. Grafos

Matematicamente, um grafo formaliza relações de interdependência existentes entre os elementos de um conjunto [Goldbarg 2012]. Em geral, é uma abstração que permite representar o relacionamento entre pares de elementos. Um grafo G = (V, E) é um conjunto de V vértices e E arestas, sendo cada aresta unindo um par de vértices. Podendo ser aplicada em diversas áreas do conhecimento. [Silva 2015]

Um exemplo simples de grafo é representado na Figura 1, no qual os pontos são os

vértices, que poderiam ser pessoas, cidades, páginas web e outros. As linhas representam arestas que poderiam ser distância, conexão, custo e assim por diante. Dependendo da aplicação, as arestas podem ser direcionadas, e são representadas por setas.

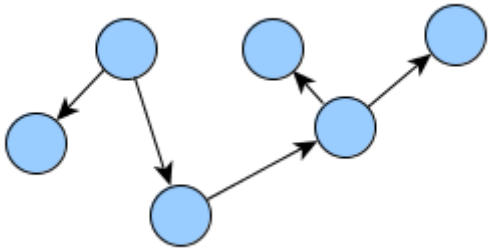

Figura 1. Exemplo de um grafo.

2.1. Banco de dados orientado a grafos

O banco de dados guiado a grafos funciona basicamente conforme o modelo que foi explicado anteriormente, porém com algumas particularidades. As entidades são vértices ou nós, e as arestas ou caminhos com a qual a entidade se relaciona são chamados de relacionamento que é representado na Figura 2.

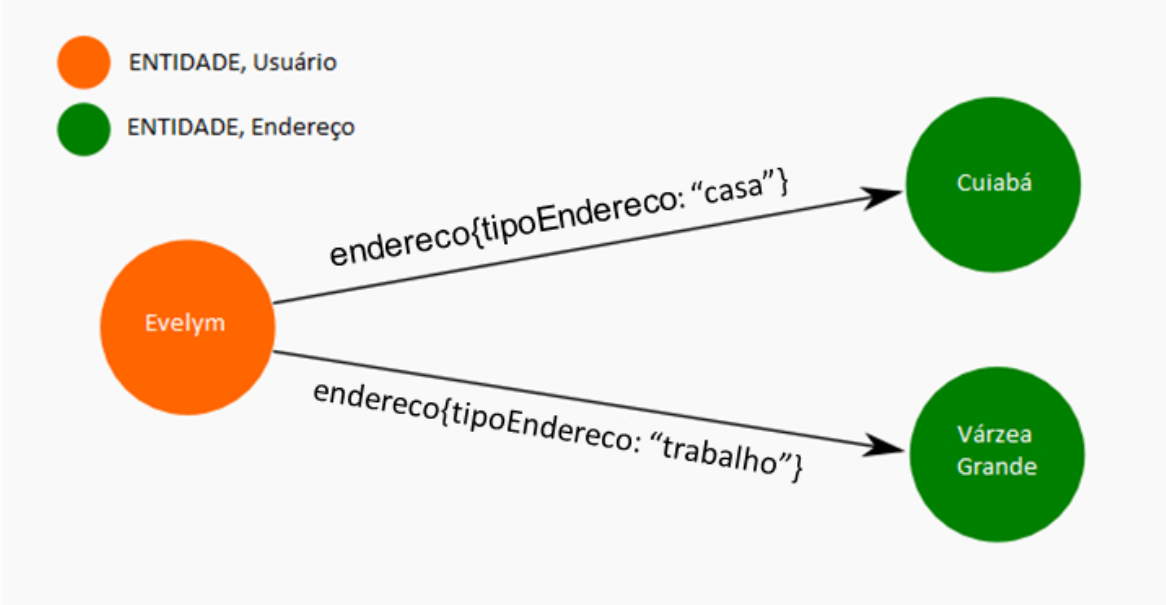

Figura 2. Exemplo de modelagem de um banco de dados orientado a grafos.

Um banco de dados orientado a grafos pode ser utilizado em mapas para redirecionar ou traçar rotas de um ponto a outro, ou até mesmo facilitar a logística de entrega de produtos, como no problema do caixeiro-viajante, que consistia em fazer entregas em diferentes cidades, traçando rotas para passar por todas as cidades utilizando o melhor caminho e tendo economia de combustível. Nas redes sociais para fazer recomendação e sugestão de usuários, podendo ser utilizado na gestão de dados governamentais para análise de impacto ou até mesmo a detecção de fraude.

O antigo modelo relacional são fracos ao manusear relacionamento entre dados, pois, eles fazem esquemas rígidos dificultando a inclusão de diferentes conexões ou adaptações para as novas requisições de negócio. O banco de dados em grafos não é apenas eficiente para o relacionamento de dados, eles também são flexíveis quando um modelo de dados é expandido ou conforme a necessidade de mudança da regra de negócio. A mudança da indústria de *software* para a adoção da nova tecnologia de banco de dados resultou em melhores produtos e a notável experiência do usuário exemplificado anteriormente.

Segundo [Robinson et al. 2015], banco de dados em grafos são geralmente construídos para uso com sistemas de transações (OLTP). Consequentemente eles são normalmente otimizados para desempenho, e projetado com integridade transacional e disponibilidade operacional. Há no mercado diversos bancos de dados baseado na estrutura de grafo, cada um deles explora diferentemente o potencial de cada característica citadas anteriormente. Pode ser visto a seguir alguns desses bancos e quais características cada um melhor explora:

- Neo4J: Banco de dados orientado a grafo de código aberto altamente escalável que oferece suporte a ACID, possui *clustering* de alta disponibilidade para implantações corporativas e vem com uma ferramenta de administração baseada na web que inclui suporte a transações completo e modelo de dados baseado em relacionamento de nó. O Neo4j é acessível a partir da maioria das linguagens de programação usando a interface da API da Web REST incorporada, bem como um protocolo proprietário Bolt com *drivers* oficiais [Neo 2016].
- AllegroGraph: Este é um moderno banco de dados gráfico, de alto desempenho e persistência. Possui um uso eficiente da memória em combinação com o armazenamento em disco, permitindo que ele alcance bilhões de quarto de um byte enquanto mantém um desempenho superior. Ele também suporta SPARQL, RDFS ++ e Prolog dotado de razões de vários aplicativos cliente, é projetado para armazenar RDF triplos, um formato padrão para dados vinculados. Um navegador personalizado, Gruff, está disponível para visualizar os grafos [All 2016].
- OrientDB: Este banco de dados é orientado a grafos distribuídos de 2ª geração, possui código aberto com a flexibilidade de documentos em um produto isto é, possui suporte nativo também a NoSQL. É um banco de dados de grafo altamente escalável com suporte a ACID completo. Ele tem uma replicação *multi-master* e *sharding*. Suporta modos sem esquema, esquema-cheio e esquema-misturado. Tem um sistema forte de perfil de segurança baseado no usuário e nas funções. Suporta uma linguagem de consulta que é tão semelhante ao SQL, ou seja diminuindo a curva de aprendizado daqueles que já dominam o conhecimento na base de dados relacional [Ori 2016].
- DEX/Sparksee é um sistema de gerenciamento de banco de dados de alto desempenho e escalável. As principais características é o seu desempenho de consulta para a recuperação e exploração de grande rede. O Sparksee mobile 5 é o primeiro banco de dados baseado em grafos para dispositivos móveis [DEX 2016].
- O Titan é um escalável e otimizado banco de dados orientado a grafos para armazenar e consultar grafos contendo centenas de bilhões de nós e relacionamentos distribuídos em um cluster multi-máquina. Os recursos incluem escalabilidade elástica e linear para uma crescente base de dados e usuários, distribuição

e replicação de dados para desempenho e tolerância a falhas, alta disponibilidade de multi-datacenter e hot backups e suporte para ACID e eventual consistência [Tit 2016].

- FlockDB é um banco de dados distribuído para armazenar listas de adjacência, com objetivos de suportar uma alta taxa de operações de inclusão/atualização/remoção, complexas consultas aritméticas, paginação através de conjuntos de resultados de consulta contendo milhões de entradas, capacidade de ”arquivar”e posterior restauração do nó, escala horizontal incluindo replicação e migração de dados online [Flo 2016].
- InfiniteGraph é um banco de dados orientado a grafo altamente especializado, ou seja, um produto comercial distribuído e habilitado para a nuvem [Inf 2016]

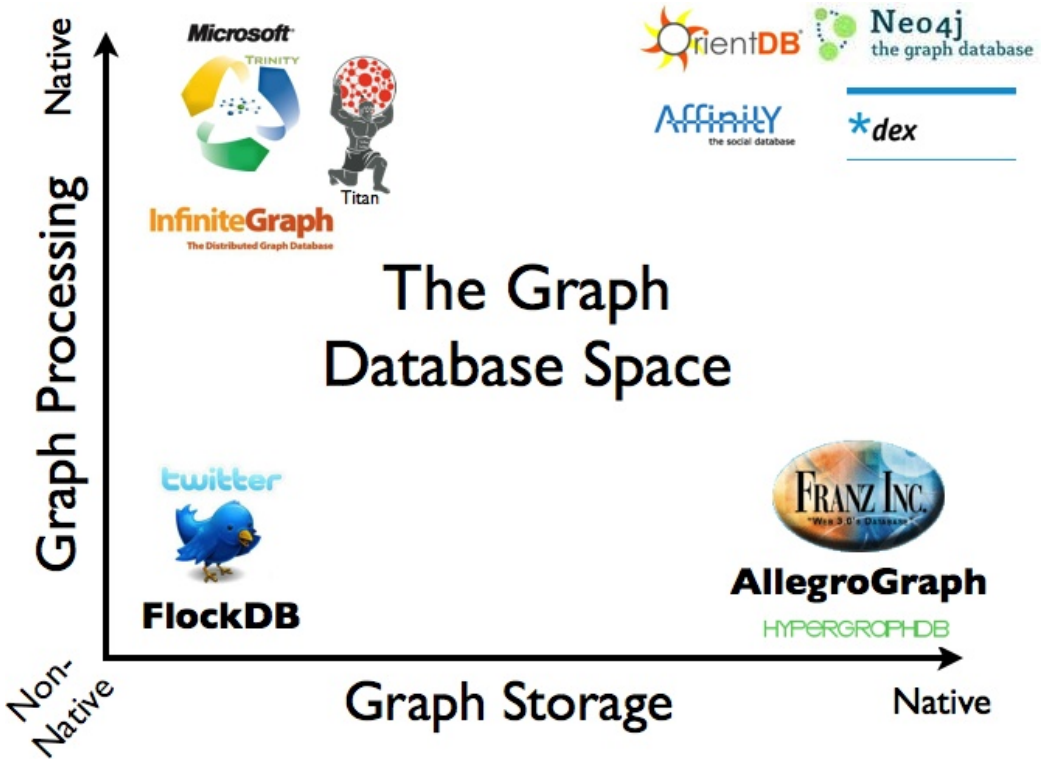

Figura 3. Uma visão geral do espaço dos bancos de dados orientados a grafos [Robinson et al. 2015]

Há duas propriedades do banco de dados orientados a grafos que podem vir nativamente, como o armazenamento subjacente e motor de processamento, como mostrado na Figura 5. O armazenamento subjacente quando é nativo, faz com que o banco seja otimizado e projetado para armazenar e gerenciar grafos. Nem todos os bancos de dados orientados a grafos possuem esse recurso nativo. Alguns fazem o seu armazenamento ao serializar os dados dos grafos em um banco de dados relacional, um banco de dados orientado a objeto, ou alguma outra proposta genérica de armazenamento de dados.

O motor de armazenamento em algumas definições requerem que o banco de dados orientados a grafos use adjacência livre de índice, isso significa conectar os nós fisicamente “ponto-a-ponto”, beneficiando uma performance transversal. Entretanto, ao fazer consultas que não usem recursos transversais ela se torna mais custosa para o banco.

Conforme o gráfico da Figura 5 a tecnologia que possui propriedades nativas de armazenamento e processamento é o Neo4J, OrientDB, Dex e o Affinity. As particularidades de cada banco orientado a grafo foram exploradas anteriormente, e cada um possui suas utilidades, como, por exemplo, o banco orientado a grafo Dex que é mais voltado para dispositivos móveis. A escolha do banco de dados orientado a grafos no caso de uso deste trabalho é o Neo4J, por ser um banco *open source* e possuir propriedades de armazenamento nativa, sendo assim, não é necessário outro tipo de banco de dados. Além dele possuir um motor de processamento melhor, pois não se tem dependência com outra tecnologia.

2.2. Neo4j

O Neo4J é um banco de dados baseado em grafos desenvolvido pela empresa Neo tecnologia, Inc. Este banco de dados é um *software* livre e possui uma vasta e ativa comunidade que o mantém. Por possuir armazenamento e processamento de dados nativo em seu núcleo, ele detém uma alta performance de desempenho. Além disso, o Neo4J é um banco altamente escalável, robusto e de fácil aprendizado, o que o torna simples de ser usado, como por exemplo na questão do modelo de dados, que por ser baseado em grafos de relacionamentos, seu modelo lógico é exatamente igual ao modelo físico.

Ainda existe a vantagem de possuir *drivers* disponíveis em diversas linguagem de programação, como: Java, Python, .Net e Javascript [Neo 2016]. Isso faz com que grandes empresas que procuram soluções baseadas na tecnologia que ele oferece, o utilizem como grande aliado no processamento de suas informações, tais como: Cisco, Ebay, Walmart, Linkedin, dentre outras.

A consulta dos dados no Neo4J é feito como em todos os banco de dados orientados a grafos, por meio de conceitos matemáticos vindo da teoria dos grafos e é utilizado como um poderoso e eficiente motor para consulta de dados. Este conceito é o grafo transversal que é a operação de visitar a configuração "ponto-a-ponto" conectado com relacionamentos, e é uma das principais formas que fazem o Neo4J tão poderoso [Vukotic et al. 2015].

A linguagem utilizada para se trabalhar com o Neo4J é a Cypher, que assim como o SQL é declarativa, mas para grafos. A estrutura da linguagem, com relação aos comandos chaves, é parecida com os comandos da linguagem SQL, como: *Where*, *Order by*, *Limit* e *And*. Entretanto, por a Cypher ser orientada a relações, diferenças existem na capacidade desta linguagem ser capaz de recuperar dados que possuem relação entre si de maneira muito mais simples do que a linguagem SQL no banco relacional. No exemplo a seguir, é mostrado uma consulta baseada nesta linguagem.

```
MATCH (p:Product)
WHERE p.productName = "Chocolade"
RETURN p.productName, p.unitPrice;
```

A consulta mostrada está selecionando do nó de produto, todos os itens que tenham nome igual a *chocolate* e retornando o seu nome e preço. Em SQL não seria muito diferente. Por outro lado, se for imaginado um cenário de uma organização empresarial em que há departamentos, projetos e membros, as consultas podem ficar mais complexas, como a SQL a seguir que tenta obter a lista de trabalhadores que estão no departamento de tecnologia.

```
SELECT nome FROM Pessoa
LEFT JOIN Pessoa_Departamento
  ON Pessoa.Id = Pessoa_Departamento.PessoaId
LEFT JOIN Departamento
  ON Departmento.Id = Pessoa_Departamento.DepartamentoId
WHERE Departamento.nome = "Departamento de Tecnologia"
```

A mesma consulta em Cypher seria:

```
MATCH (p:Pessoa)<-[:Trabalha]-(d:Departamento)
WHERE d.nome = "Departamento de Tecnologia"
RETURN p.nome
```

Portanto, em se tratando de dados que possuem forte relações, é mais simples e menos custoso construir uma consulta em Cypher, além do banco de dados já estar otimizado para resolver esse tipo de consulta.

3. Caso de Estudo

Este trabalho analisou o desenvolvimento de uma estrutura de difusão de conhecimento, promovida no âmbito da Revista Brasileira de Meteorologia [RBM 2016], e como se comportam as ligações entre os atores envolvidos. Os dados analisados foram os artigos publicados na revista RBMET, entre janeiro e maio de 2014, cujo autores foram considerados os atores da rede de relações, e as ligações entre esses atores a estrutura de colaboração desta rede na produção acadêmica analisada. Para tanto, foi feito antes a obtenção, preparação e importação dos dados destes artigos. Em seguida a utilização dessa massa de dados no banco Neo4j.

3.1. Obtenção dos dados

Na captura dos dados foi utilizado um *webcrawler* que é uma rotina automatizada de computador que salva em arquivo as informações de páginas *web* repassadas a ele para análise, exceto aquelas que são protegidas para não aceitarem esse tipo de rotina. No apêndice A deste trabalho foi incluído o código fonte do *webcrawler* utilizado para obtenção dos dados. Seu funcionamento consiste em encontrar informações que estão separadas por marcações dentro do código fonte da página, assim a rotina extrai as informações contidas nessas marcações e as escreve em arquivo texto com padrão pré-formatado. Com isso foi possivel obter uma amostra relevante para o uso neste trabalho. Como suplemento ao crawler foi realizada uma segregação dos dados utilizando de planilhas de dados, conforme mostrado na figura 4.

3.2. Preparação dos dados

Os dados relacionados ao objeto de estudo foram analisados numa rotina automatizada construída para segregar os autores, palavras-chave e artigos. Esta rotina também fez a preparação dos dados, que consistiu na capacidade de verificar similaridades das informações. Essa construção da base de dados iniciou da especificação dos autores e dos artigos, seguida do estabelecimento das ligações entre os atores (autores).

A codificação dos autores foi feita segundo seus sobrenomes e nomes e respectivos e-mails, de forma a contornar o problema de duplicidade com o uso exclusivo da

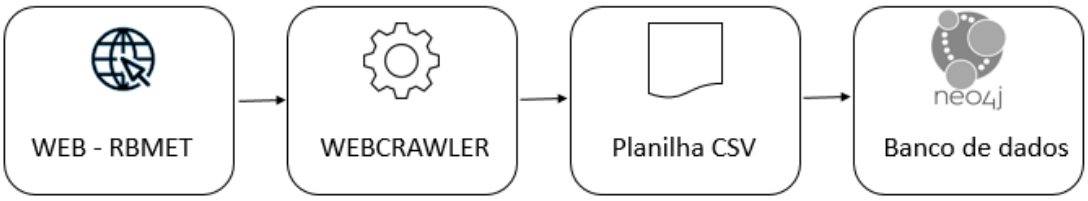

Figura 4. Fluxograma da obtenção dos dados

combinação dos sobrenomes e nome. Para facilitar a busca de autores idênticos foi necessário que o autor tivesse um número de identificação, que quando a rotina detecta a duplicidade deste autor este é relacionado com o identificador do artigo.

Os e-mails não foram o principal mecanismo de identificação de autores, afinal o e-mail pode ter vínculo com uma determinada instituição, e além disso os autores podem lecionar em mais de uma instituição ou por terem migrado de uma instituição para outra. Dessa forma, cada autor pode encontrar-se identificado com nenhum, um ou vários e-mails.

Ao analisar as palavras-chaves dos artigos verificou a necessidade de também criar o mesmo mecanismo para identificação de duplicidades e assim relacioná-las com outros trabalhos acadêmicos. A problemática em filtrar palavras-chaves ocorreu devido a sua falta de padronização. Mesmo optando pelo idioma inglês, foi necessário a normatização sem a acentuação.

Em um primeiro momento, as ligações entre os autores e palavras-chaves foram estabelecidas pela listagem adequada de quem eram os autores e as palavras-chaves relacionados com o trabalho apresentado, já devidamente codificados e com o número de seus vértices associados. A numeração dos autores e palavras-chaves foi atribuída por sua ordem de aparição na relação de artigos apresentados, não obedecendo a nenhum critério hierárquico de importância.

3.3. Importação dos dados

Após o procedimento relatado acima, a rotina automatizada gerou arquivos que organizou as informações de autores, palavra-chave, artigo e arquivos de relacionamento. Primeiramente foi executado os comandos de criação dos dados. A seguir é mostrado a instrução Cypher de importação:

```
LOAD CSV WITH HEADERS FROM
"file:///Artigo.csv"
AS csvLine FIELDTERMINATOR  ';'
CREATE (a:Artigo
{
id: toInt(csvLine.id),
titulo:csvLine.titulo,
ano:csvLine.ano
})
```

```
LOAD CSV WITH HEADERS FROM
"file:///Autor.csv"
AS csvLine FIELDTERMINATOR  ';'
CREATE (b:Autor
{
id: toInt(csvLine.id),
nome:csvLine.nome,
email: csvLine.email
})

LOAD CSV WITH HEADERS FROM
"file:///PalavraChave.csv"
AS csvLine FIELDTERMINATOR  ';'
CREATE (c:PalavraChave
{
id: toInt(csvLine.id),
descricao:csvLine.descricao
})
```

Com os dados persistidos no banco Neo4j, foi executado um comando de verificação dos identificadores. Este comdando verificou se é único o campo de identificação, para que ao executar instruções de correlação não haja conflitos.

```
CREATE CONSTRAINT ON (artigo:Artigo)
ASSERT artigo.id IS UNIQUE

CREATE CONSTRAINT ON (autor:Autor)
ASSERT autor.id IS UNIQUE

CREATE CONSTRAINT ON (palavrachave:PalavraChave)
ASSERT palavrachave.id IS UNIQUE
```

Foi feito o mapeamento das relações existentes entre os artigos por meio de seus atributos. O intuito com tal procedimento foi identificar atributos comuns entre os artigos inseridos na base, para que fosse possível consultas flexíveis aos vínculos existentes.

```
USING PERIODIC COMMIT 500
LOAD CSV WITH HEADERS FROM
"file:///ArtigoAutor.csv"
AS csvLine FIELDTERMINATOR  ';'
MATCH (artigo:Artigo
{id: toInt(csvLine.artigoId)}),
(autor:Autor
{id: toInt(csvLine.autorId)})
CREATE (artigo)-[:ESCRITO]->(autor)

USING PERIODIC COMMIT 500
LOAD CSV WITH HEADERS FROM
"file:///ArtigoPalavraChave.csv"
```

```
AS csvLine FIELDTERMINATOR  ';'
MATCH (artigo:Artigo
{id: toInt(csvLine.artigoId)}),
(palavrachave:PalavraChave
{id: toInt(csvLine.palavrachaveId)})
CREATE (artigo)-[:PalavraChave]->(palavrachave)
```

3.4. Uso do Neo4j

Os resultados obtidos foram analisados com o objetivo de levantar as vantagens e desvantagens proporcionadas com o uso de um banco orientado a grafos no cenário proposto. Dessa maneira demonstrando as características técnicas do banco que contribuíram para o ganho de seu uso, assim como apresentando as dificuldades encontradas no emprego deste modelo de banco de dados no cenário proposto.

Foram importadas no total 59 artigos, 192 autores e 230 palavras-chave. Após o mapeamento das relações dos artigos presentes na base, foram identificados no total 214 relacionamentos entre artigos com autores e 236 relacionamentos entre palavras-chaves com artigo.

A ferramenta disponibilizada pelo Neo4j possui um aspecto visual que facilita como as informações são mostradas. Dessa forma, é mais fácil olhar o relacionamento das informações. As informações são disponibilizadas no formato de nó que possui várias arestas que são os relacionamentos, e para nós de informações diferentes são disponibilizados de cores diferentes.

Por fim, as instruções em Cypher são bem parecidas com as SQL o que facilita o aprendizado da linguagem, porém ainda mais simples e com algumas particularidades ao fazer consultas com relacionamentos.

4. Resultados

Foram planejadas três tipos de consultas que fazem parte do objetivo deste trabalho: qual artigo tem relação com outro artigo; qual autor poderia avaliar um determinado artigo; quais autores têm conflito de interesse com outro autor.

Para o primeiro caso, pode ser utilizado uma consulta genérica no qual dois artigos que contenham uma palavra-chave em comum é um indício da relação entre os dois artigos.

Esta consulta pode nos trazer vários resultados, mas para delimitar o contexto analisado foi selecionado o identificador de número 6, que seu título é *Incident and reflected Photosynthetically Active Radiation above and below canopy in the Mata Atlantica forest in Coruripe, Alagoas*. O resultado desta consulta é demonstrado na Tabela1. Foram retornadas as informações mais relevantes, como o título de cada artigo e a palavra-chave que é o nó dessa ligação, assim como um contador de palavras-chave. Logo após foi feita uma ordenação da descrição do título dos artigos e respectivamente por sua quantidade em ordem decrescente, como mostrado abaixo no código da consulta realizada.

```
MATCH (a1:Artigo)-->(p:PalavraChave)<--(a2:Artigo)
WHERE a1.id <> a2.id
AND a1.id = 6
```

```
RETURN a1.titulo AS Título, a2.titulo AS Artigo_Relacionado,
p.descricao AS Palavra_Chave,
count(*) as Qtd
ORDER BY p.descricao, numArtigos DESC
```

Artigo	Artigo relacionado	Palavra chave	Qtd
Incident and reflected Photosynthetically Active Radiation above and below canopy in the Mata Atlantica forest in Coruripe, Alagoas	Assesment of energy balance usint TM - Landsat 5 digital images and SEBAL algorithm over the southern Pernambuco seacost	vpm model	1
Incident and reflected Photosynthetically Active Radiation above and below canopy in the Mata Atlantica forest in Coruripe, Alagoas	Impacts of climate change in the ecoclimatology of Aleurocanthus Woglumi Ashby, 1903 (Hemiptera: Aleyrodidae) in the state of Pará	vpm model	1
Incident and reflected Photosynthetically Active Radiation above and below canopy in the Mata Atlantica forest in Coruripe, Alagoas	Energy balance and evapotranspiration of the grass Mombaça under rotated pasture	vpm model	1

Tabela 1. Artigo tem relação com outro artigo.

Assim como na consulta de dados mostrada anteriormente, pode ser consultado também a relação entre autores. Nos artigos encontrados na base foi possível verificar quais autores trabalharam juntos, ou seja, autores que possuem conflitos de interesse em caso de avaliação. A consulta demonstrada no código abaixo consiste encontrar o autor e os coautores de um determinado artigo, no exemplo demonstrado na Tabela 2 utilizamos o autor de identificador 25, chamado Marcos Antonio Lima Moura. Por meio da relação foi retornado como resultado o nome do autor e seus respectivos coautores.

```
MATCH (autor2:Autor)<--(artigo1:Artigo)-->(autor1:Autor)
WHERE autor1.id <> autor2.id
AND autor1.id = 25
RETURN distinct autor1.nome, autor2.nome
```

Autor	Autor Relacionado
Marcos Antonio Lima Moura	Rosiberto Salustiano da Silva Junior
Marcos Antonio Lima Moura	Aurilene Barros dos Santos
Marcos Antonio Lima Moura	Rayonil Gomes Carneiro
Marcos Antonio Lima Moura	Antonio Marcos Delfino de Andrade

Tabela 2. Autores que trabalharam juntos.

Pode ser ido além ao realizar uma consulta para buscar qual autor pode avaliar um determinado artigo. A busca consiste em identificar quais artigos possuem uma relação através da palavra chave, e que não tenha relações com autores de outros artigos. No exemplo mostrado como resultado na Tabela 3 foi utilizado o artigo com identificador 6 que seu título é *Incident and reflected Photosynthetically Active Radiation above and below canopy in the Mata Atlantica forest in Coruripe, Alagoas* e o seu autor, chamado Marcos Antonio Lima Moura, com a identificação 6. A tabela de resultados mostra o nome dos autores que podem avaliar o artigo com a quantidade de artigos com palavra-chave analisada.

```
MATCH (autor1:Autor)<--(a1:Artigo)-->(p:PalavraChave)
<--(a2:Artigo)-->(autor2:Autor)
WHERE a1.id <> a2.id
AND a1.id = 6
AND autor1.id = 25
AND NOT (autor1:Autor)<-[:escrito]-(a1:Artigo)-[:escrito]->
(autor2:Autor)
RETURN autor2.nome, count(*) as numArtigosComPalavraChave
ORDER BY numArtigosComPalavraChave DESC
```

Autor	Qtd
Everaldo Barreiros de Souza	2
Barbara dos Santos Esteves	1
João Batista Miranda Ribeiro	1
Elias Fernandes de Sousa	1
Rodrigo Almeira Muniz	1
Douglas Batista da Silva Ferreira	1
Bergson Cavalcanti de Moraes	1
Manoel Bandeira de Albuquerque	1
Bernardo Barbosa da Silva	1
Josiclêda Domiciano Galvíncio	1
Célia Cristina Machado	1
Lidiane de Lima Lousada	1
José Carlos Mendonça	1

Tabela 3. Autores poderiam avaliar um determinado artigo.

O banco de dados orientado a grafo possui uma visão ampla dos dados onde é possivel visualizar as consultas feitas em um aspecto maior demonstrado na Figura 5.

5. Conclusões

Este trabalho avaliou a aplicabilidade de um mecanismo de relacionamento de dados de artigos acadêmicos. Para essa avaliação foram levantados a busca de dados, que foram extraidos da RBMET, organizados e armazenados em um banco de dados orientado a grafos, o Neo4j. Com isso foi possivel responder os principais questionamentos desta pesquisa: quais autores tem conflito de interesse com outro autor, ou qual autor poderia auxiliar no trabalho de outro.

Os resultados obtidos neste artigo, embora exploratórios, conseguiu atingir o seu objetivo final que sugere um forte embasamento em redes de colaboração entre artigos. Pode ser utilizado no meio acadêmico como sugestão e apontamento para outro tipo de consultas ou até mesmo mineração de dados, difundindo assim o conhecimento nas instituições de ensino. Finalmente, sugere-se a utilização de métodos de pesquisa complementares, como a ampliação do universo amostral das informações incluídas no banco.

Como trabalhos futuros pode ser feito um estudo de caso implantando a tecnologia em um sistema com acesso total à base de dados de uma revista, tendo acesso a outros atributos dos artigos, aumentando o nível de informação dos relacionamentos. Dessa forma, será possível auxiliar pesquisadores a correlacionar dados de suas pesquisas.

Referências

(2016). Allegrograph. `http://franz.com/agraph/allegrograph/`. acessado em 04/12/2016.

(2016). Dex/sparksee. `http://sparsity-technologies.com/`. acessado em 04/12/2016.

(2016). Flockdb. `https://github.com/twitter/flockdb`. acessado em 04/12/2016.

Figura 5. Uma visão geral das consultas realizadas.

(2016). Infinitegraph. http://www.objectivity.com/products/infinitegraph/. acessado em 04/12/2016.

(2016). Newo4j. https://neo4j.com/top-ten-reasons/. acessado em 04/12/2016.

(2016). Orientdb. http://orientdb.com/orientdb/. acessado em 04/12/2016.

(2016). Rbmet. http://www.scielo.br/scielo.php?script=sci_serial&pid=0102-7786&lng=en&nrm=iso. acessado em 04/12/2016.

(2016). Titan. http://titan.thinkaurelius.com/. acessado em 04/12/2016.

Araujo, R. M., Muramatsu, T. Y., Silveira, B. A., and Revoredo, K. (2015). Minerando publicações científicas para análise da colaboração em comunidades de pesquisa-o caso da comunidade de sistemas de informação/mining scientific publications to analyse the collaboration in research communities-the case of the information systems community. *Revista Electronica de Sistemas de Informaçao*, 14(1):1.

Goldbarg, M. (2012). *Grafos: Conceitos, algoritmos e aplicações*. Elsevier Brasil.

Mena-Chalco, J. P., Digiampietri, L. A., and Cesar-Jr, R. M. (2012). Caracterizando as redes de coautoria de currículos lattes. In *Brazilian Workshop on Social Network Analysis and Mining (BraSNAM)*, pages 1–12.

Robinson, I., Webber, J., and Eifrem, E. (2015). *Graph Databases: New Opportunities for Connected Data*. "O'Reilly Media, Inc.".

Silva, A. S. (2015). O teorema de euler e algumas aplicações.

Vukotic, A., Watt, N., Abedrabbo, T., Fox, D., and Partner, J. (2015). *Neo4j in Action*. Manning.

A. Apêndice

Este apêndice demonstra o código do *web crawler* utilizado na obtenção dos dados.

```
# -*- coding: utf-8 -*-
import urllib
import re


f = urllib.urlopen("http://dontpad.com/revbrasmet")
dontpad = f.read()
idx = dontpad.find("<textarea id=")
idx2 = dontpad.find("</textarea>")

for link in dontpad[idx+20:idx2-1].splitlines():
        link = link.replace("?=", "?script=")
        f = urllib.urlopen(link)
        content = f.read()

        indiceProcuraUrl = 0;

        while (1):

                idx = content.find("<font face=\"Symbol\"
                 ↪ color=\"#000080\">· </font><a
                 ↪ href=\"http://www.scielo.br/scielo.php?script=sci_a
                 ↪ indiceProcuraUrl)
                indiceProcuraUrl = idx;

                if idx == -1:
                        break;

                else:
                        idx2 = content.find("\">text in",
                         ↪ idx)
                        indiceProcuraUrl = idx2;

                        url =
                         ↪ content[idx+55:idx2].replace("&amp;",
                         ↪ "&")
                        f2 = urllib.urlopen(url)
                        artigo = f2.read()

                        idxTitulo =
                         ↪ artigo.find("citation_title") +
                         ↪ 25
```

```
35                      idxTitulo2 = artigo.find("\">",
                         ↪ idxTitulo)
36                      titulo =
                         ↪ artigo[idxTitulo:idxTitulo2]
37
38                      idxAutores =
                         ↪ artigo.find("citation_author")
                         ↪ + 26
39                      idxAutores2 = artigo.find("\">",
                         ↪ idxAutores)
40                      autor =
                         ↪ artigo[idxAutores:idxAutores2]
41
42                      email =
                         ↪ re.findall(r'[\w\.-]+@[\w\.-]+',
                         ↪ artigo)
43
44                      idxAno =
                         ↪ artigo.find("citation_date") +
                         ↪ 24
45                      idxAno2 = artigo.find("\">",
                         ↪ idxAno)
46                      ano = artigo[idxAno:idxAno2]
47
48                      idxKeywords = artigo.find("Key
                         ↪ words:") + 15
49                      idxKeywords2 = artigo.find("</p>",
                         ↪ idxKeywords)
50                      if idxKeywords == 14:
51                              idxKeywords =
                                 ↪ artigo.find("Keywords:")
                                 ↪ + 14
52                              idxKeywords2 =
                                 ↪ artigo.find("</font>",
                                 ↪ idxKeywords) - 1
53                      keywords =
                         ↪ artigo[idxKeywords:idxKeywords2]
54                      keywords = keywords.replace(";",
                         ↪ ",");
55
56                      print titulo, ";", autor, ";",
                         ↪ keywords, ";", email, ";", ano
```