Universidade Federal de Pernambuco

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CENTRO DE INFORMÁTICA

2011.2

Otimização de uma Ferramenta para Sumarização de Ontologias

Trabalho de Graduação

Aluno Paulo Orlando Vieira de Queiroz Sousa {povqs@cin.ufpe.br}

Orientador Ana Carolina Salgado {acs@cin.ufpe.br}

13 de Dezembro de 2011

Universidade Federal de Pernambuco

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CENTRO DE INFORMÁTICA

2011.2

Otimização de uma Ferramenta para Sumarização de Ontologias

Trabalho de Graduação

Trabalho de graduação apresentado no Centro de Informática da Universidade Federal de Pernambuco por Paulo Orlando Vieira de Queiroz Sousa, orientado por Ana Carolina Salgado, como requisito para a obtenção do Grau de Bacharel em Ciência

Aluno Paulo Orlando Vieira de Queiroz Sousa {povqs@cin.ufpe.br}

Orientador Ana Carolina Salgado {acs@cin.ufpe.br}

13 de Dezembro de 2011

Folha de Aprovação

Otimização de uma Ferramenta para Sumarização de Ontologias

Paulo Orlando Vieira de Queiroz Sousa

Aprovado em 15 de Dezembro para ser apresentado.

Ciente:

_____

Profª. Ana Carolina Salgado, PhD – UFPE (Orientadora)

Agradecimentos

A Deus por ter me dado forças e iluminando meu caminho para que pudesse concluir mais uma etapa da minha vida.

Em especial, gostaria de agradecer aos meus pais por todo o carinho e dedicação durante todos os momentos da minha vida, sempre mostrando que todos os objetivos da vida podem ser alcançados através de trabalho e dedicação.

Não poderia deixar de ser grato: a professora Ana Carolina, que sempre esteve presente transmitindo confiança e apoio nos momentos importantes da minha formação acadêmica, orientando e incentivando o meu crescimento; a Carlos que me auxiliou na iniciação cientifica de todas as formas e a todos os membros do projeto SPEED que contribuíram de diferentes formas na realização desse trabalho.

Agradeço, também, a todos os amigos que contribuíram durante o andamento desse trabalho ou na passagem da graduação.

Resumo

Mais e mais ontologias têm sido desenvolvidas e aplicadas em diferentes abordagens, tais como, Web Semântica, Integração de Dados e Inteligência Artificial. A ontologia em uma simples definição representa um conhecimento, ou seja, representa de forma explícita as especificações formais de um domínio. Assim sendo, para o uso adequado de uma ontologia faz-se necessário compreender a representação do seu conhecimento. No intuito de ajudar nesse entendimento, analisou-se formas para fornecer maior velocidade e melhor objetividade de leitura em uma ontologia. Com esse propósito, pensou-se em maneiras de sumarizar uma ontologia, destacando as partes importantes em um subconjunto de conceitos que represente de forma resumida o conhecimento da ontologia original.

Este trabalho propõe desenvolver uma ferramenta para sumarização automática de ontologias, baseando-se na ferramenta OWLSum [1,2,3]. A ferramenta desenvolvida neste trabalho é uma evolução da OWLSum, que manterá as mesmas funcionalidades, tais como, permitir visualização da ontologia em forma de grafo, utilizar medidas de centralidade e freqüência para definir a relevância dos conceitos e gerar sumários baseado nos parâmetros do usuário. Além do mais, terá um novo algoritmo que possibilitará sumarizar a ontologia, realizando uma busca por conceitos mais relevantes e próximos; e duas novas medidas para avaliar a relevância, baseado na aproximação de conceitos relevantes e na nomenclatura do conceito. O presente documento mostrará um estudo sobre sumarização de ontologia e um comparativo entre as duas versões da ferramenta.

Palavras-chave: Ontologia, Sumarização de Ontologia

Sumário

Índice de Figuras

Índice de Tabelas

1. Introdução

Ontologia, na filosofia, é a ciência que estuda o ser, definindo “o que é” dos tipos de estrutura dos objetos, propriedades, eventos, processos e relações em cada área presente na realidade [4]. Na definição de Gruber [5], ontologia é uma especificação formal explícita de uma conceituação compartilhada. Já do ponto de vista da engenharia do conhecimento as ontologias incorporam conhecimentos formalizados que podem ser compreendidos e reutilizados [6].

Além dessas definições, a ontologia pode ser um padrão de linguagem, baseada na lógica de descrição, que foi projetada para estruturar representações de conhecimento através de relacionamentos e correspondências semânticas [7]. O uso da ontologia abrange várias aplicações tais como: inteligência artificial, através do raciocínio dedutivo; representação de conhecimento, com relacionamento semântico; e padronização de comunicação, com a estrutura em XML.

Através da capacidade de representar o conhecimento, a ontologia tem desempenhado um papel central no desenvolvimento e implantação da Web Semântica. Além do mais, com o surgimento de novas aplicações semânticas, a Web Semântica ganha destaque, possibilitando publicar e compartilhar um número maior de modelos de conhecimento formalizado [8]. O compartilhamento dos modelos em conjunto com a capacidade de reusar informações semânticas tem auxiliado na produção de novas ontologias, incentivando os desenvolvedores, primeiramente, a investigar e compreender os modelos existentes em vez de desenvolver um novo do zero [6]. Tais processos, reuso e desenvolvimento, exigem conhecimento especializado para serem exercidos.

O entendimento de ontologia é significativo no processo de reuso e desenvolvimento. Uma abordagem para auxiliar no entendimento humano é a representação gráfica da estrutura da ontologia em um grafo direcionado [9], que representa os conceitos em vértices, as relações em arestas e a direção no sentido da ação. No entanto, mesmo com a visualização gráfica ainda há dificuldade de compreender ontologias complexas, pois a quantidade de conceitos e relacionamentos torna o entendimento um desafio até mesmo para especialistas. Uma solução para facilitar a compreensão e aumentar a velocidade de leitura, independente de ser homem ou máquina, é a geração de um resumo da ontologia, uma abordagem que tanto pode

auxiliar no entendimento da ontologia, para facilitar o reuso [10], quanto na otimização de leitura, para dar maior velocidade aos sistemas [3].

A sumarização de ontologia consiste em gerar automaticamente uma representação com os conceitos mais relevantes, mantendo a integridade das relações. O resumo gerado de ontologia proporcionaria: para uma máquina uma leitura rápida e objetiva dos conceitos mais importantes da ontologia original; para o homem a visualização de um recorte da ontologia original, mantendo a integridade das relações e os conceitos mais relevantes que foram definidos pelas medidas da sumarização.

Entre as obras na literatura sobre sumarização de ontologia são destacadas algumas medidas para geração de resumo. Neste trabalho utilizaram-se algumas medidas já definidas como: Centralidade, que é definida através do número de relacionamentos no conceito [2,3,6]; Freqüência, que é definida pelo número de ocorrência de um conceito em varias ontologias [1,2,3] e Simplicidade do nome, que define o grau de simplicidade do nome do conceito [11]. Também foi criada uma nova métrica chamada Proximidade, a qual se baseia na medida *closeness centrality*, que é utilizada em grafo para medir a centralidade pela aproximação entre vértices [11]. Em relação ao critério de validação do resultado foram definidas algumas métricas para avaliar a cobertura e o valor médio da relevância nos resumos gerados.

Em resumo, o atual trabalho tem o intuito de desenvolver uma ferramenta parametrizável para sumarização de ontologias, uma melhoria da ferramenta OWLSum [1,2,3]. As principais contribuições da nova versão da ferramenta denominada OWLSumBPR são: (i) geração automática de resumos de ontologias com tamanhos parametrizáveis; (ii) capacidade de gerar uma subontologia considerando os critérios de Centralidade [2,3,6], Freqüência [1,2,3], Simplicidade do nome [11] e Proximidade para garantir a participação dos conceitos mais relevantes no sumário; e (iii) uso de um novo algoritmo denominado *Broaden Paths RelevanceI*(BPR) para sumarização de ontologia.

Este documento está organizado da seguinte forma: O capítulo 2 inclui a definição e cenários para sumarização de ontologia. O capítulo 3 apresenta uma visão geral da ferramenta de sumarização. O capítulo 4 descreve as medidas utilizadas e o estudo realizado. O capítulo 5 mostra os processos dos algoritmos de sumarização. O capítulo 6 apresenta um avaliação com a versão anterior da ferramenta OWLSum [1,2,3]. O capítulo 7 apresenta a ferramenta proposta. Finalmente, no capítulo 8, concluo o trabalho e forneço uma perspectiva futura.

2. Sumarização de Ontologia

Neste capítulo apresentamos o que representa a sumarização de uma ontologia, assim como cenários onde aplicá-la.

2.1. Definição Sumarização de Ontologia

Pela definição de "resumo" em processamento de linguagem natural dada em [13], as características de resumo incluem: (i) poder produzi resumo a partir de um único documento ou vários documentos, (ii) preservar informações importantes, mantendo a coerência; (iii) gerar um texto menor, não mais do que a metade do texto original. No contexto do desenvolvimento de ontologia, a segunda característica é a que diferencia fundamentalmente a sumarização de ontologias de outras técnicas similares. Embora outras técnicas também visem reduzir o tamanho ou a complexidade da ontologia original, elas não mantêm as informações mais "importantes". É o caso de *ontology partitioning* e *ontology modularization*. A *ontology partitioning* realiza a divisão de uma grande ontologia para muitas subontologias que podem reconstituir e cobrir cada subtópico da ontologia original, além de facilitarem a utilização [14]. Já a *ontology modularization* foca no uso e na reutilização de pequenas partes que correspondem a determinados aspectos da ontologia original [15]. Além do mais, sumarização de ontologia deve ser um processo automático que Cheng et al. [6] definiram como "o processo de destilação de conhecimento da ontologia para produzir uma versão resumida para um determinado usuário (ou usuários) e tarefa (ou tarefas)".

2.2. Cenários para Sumarização de Ontologia

Um cenário típico em que a necessidade de sumarizar ontologia surge é quando um usuário tenta usar um motor de busca semântica, por exemplo, SWoogle para localizar e explorar ontologias que possam fornecer algumas conceituações relevantes para o modelo atual [16]. Em tal cenário, o usuário também pode se beneficiar nas respostas recebidas ao sumarizar os modelos retornados para conseguir uma rápida compreensão e fácil comparação entre as ontologias. Outro cenário em que a sumarização de ontologia pode ser aplicada é em um sistema de compartilhamento de dados baseado em semântica na qual utiliza a ontologia para organização e pesquisa de dados [3]. A exemplo desses sistemas há o PDMS semântico cuja arquitetura e pontos

de acessos na rede se baseiam em ontologias o qual pode utilizar sumários de ontologias para geração de índices que auxiliam na otimização e organização do sistema.

3. Visão Geral da Ferramenta

Do ponto de vista geral para a criação de uma ferramenta que sumarize uma ontologia precisa-se definir: quais os tipos de dados e arquivos a serem utilizados e como é o processo de sumarização na aplicação.

3.1. Padrão de dados

As linguagens para representação de ontologia mais aplicadas são RDF Schema e OWL. Para essa ferramenta foi adotado o padrão de ontologia OWL (Ontology Web Language), uma linguagem formal para expressar e definir ontologias [17]. Essa linguagem pode incluir descrições de classes, instâncias, propriedades e relacionamentos. Baseada na linguagem DAML (DARPA Agent Markup Language)+OIL (Ontology Interchange Language) [18], é atualmente o padrão recomendado pela W3C (World Wide Web Consortium). O padrão OWL foi projetado para melhorar a expressividade semântica e para disponibilizar uma forma comum de processar o conteúdo da ontologia. Já que a OWL é baseada em XML, a informação pode ser facilmente trocada e manipulada por diferentes tipos de sistemas. Desse modo, a escolha da linguagem OWL foi determinada pela facilidade de manipular as informações do arquivo e por ser o padrão recomendado pela W3C.

3.2. Processo de sumarização na aplicação

A Figura 1 ilustra o processo de sumarização da ferramenta que consiste em: dada uma ontologia de entrada O gerar uma versão resumida, chamando-a de ontologia sumarizada (denotado por OS). Inicialmente são calculadas as relevâncias dos conceitos, com base nos parâmetros e medidas que foram definidos pelo usuário, para formar uma hierarquia de importância entre os conceitos de O (representado por tonalidades em cinza). Em seguida é realizada a geração de OS, que corresponde a uma subontologia de O, concentrando o número máximo de conceitos de maior grau de relevância de acordo com o tamanho especificado para o resumo. Como os conceitos de maior relevância podem ser não-adjacentes em O, é possível que conceitos menos importantes (tons mais claros de cinza) sejam introduzidos em OS. Tais conceitos são

necessários para manter a integridade e preservar os relacionamentos entre os conceitos de grande relevância da ontologia original. Por isso, OS corresponde a uma subontologia de O, contendo os conceitos de maior relevância devidamente interconectados, evitando qualquer intervenção humana.

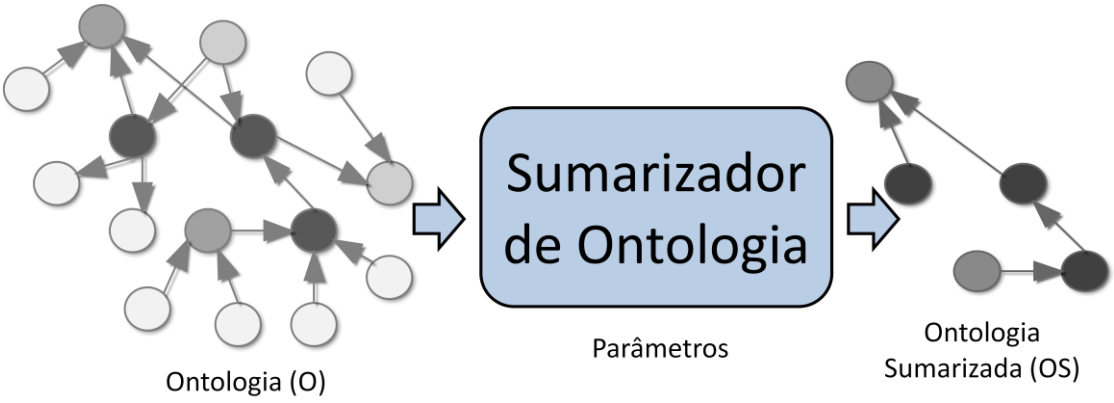

Figura 1 - Processo Geral de Sumarização de Ontologia

Na ferramenta, a ontologia *O* é modelada como um grafo direcionado com conexões rotuladas $O = (C, R)$, onde $C = (c_1, ..., c_n)$ é um conjunto finito de vértices (conceitos) e $R = (r_1, ..., r_n)$ é um conjunto finito de arestas (relacionamentos entre os conceitos). Da mesma forma, definir um resumo de uma ontologia *OS* é como criar um subgrafo de *O* no qual $OS \subset O$. Formalmente, $OS = (CS, RS)$, onde $CS \subset C$ e $RS \subset R$.

4. Medidas de Relevância na Ontologia

Nesse capítulo serão apresentados: o estado da arte de medidas de relevância aplicadas em ontologias e a descrição das medidas adotadas na ferramenta.

4.1. Estado da Arte em medidas de relevância

As medidas de avaliação de relevância são fundamentais para a sumarização de ontologia, pois são responsáveis em definir automaticamente os conceitos mais importantes na ontologia. Em função disso, realizaram-se estudos em diferentes campos tais como teoria dos grafos, categoria natural [19] e análise de rede para encontrar formas de avaliar os conceitos das ontologias.

Dentro da teoria dos grafos e análise de rede, as medidas amplamente utilizadas são: *degree centrality*, *betweenness centrality*, *closeness centrality e eigenvector centrality*. Essas medidas determinam a importância relativa de um vértice dentro de um grafo, definindo um valor de centralidade do vértice [20]. Em relação às medidas citadas, podemos destacar o trabalho de Zhang et al. [6], que realizaram um estudo comparativo com variações das medidas para aplicação na sumarização de ontologias as quais são descritas na Tabela 1:

Tabela 1 - Correspondências das medidas de centralidade

Medidas em Zhang et al.	Descrição
weighted in-degree (CI)	Medida baseada em *degree centralite* que considera o numero de relações do vértice.
betweenness centrality (CB)	Medida baseada em *betweenness centrality* que considera o numero de caminhos que passa pelo vértice.
weighed PageRank (CP) **weighted HITS (CH)** **focused weighted PageRank (CF)**	Medidas baseadas em *eigenvector centrality* que verifica a importância do vértice em função dos seus relacionamentos com outros vértices.

Na Figura 2 podemos visualizar um gráfico em que Zhang et al. calcularam valores estatísticos Kendall's tau [6], que comparam a ordem de classificação definida pelas medidas com a ordem julgada pelos especialistas, em função da probabilidade do conceito seguir um conteúdo na ontologia.

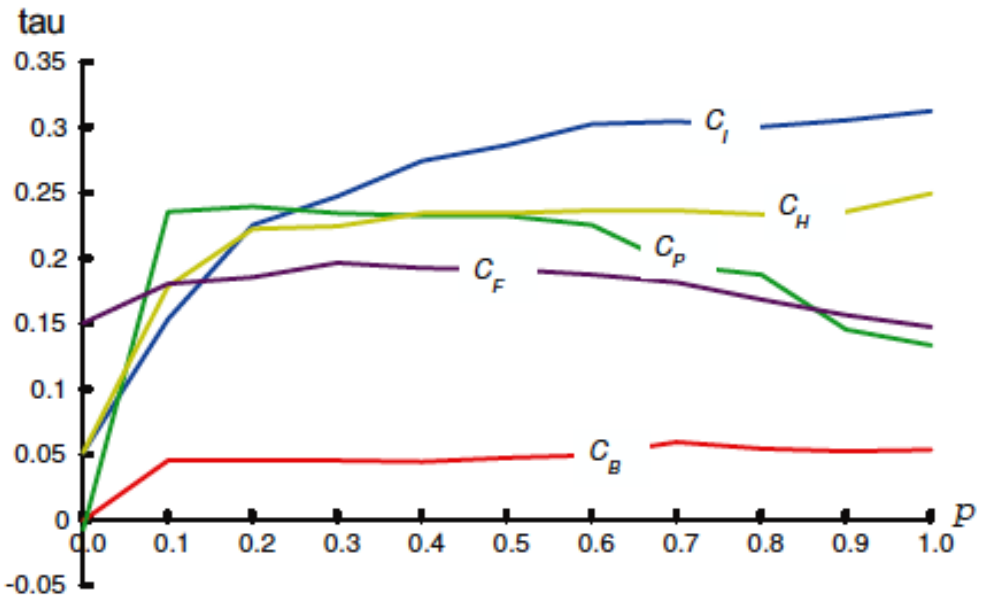

Figura 2 - Desempenho das medidas em função da probabilidade do conteúdo

O resultado mostrou que a variação da medida baseada em *degree centrality*, apesar da sua simplicidade, foi a melhor avaliada seguida das medidas que são baseadas em *eigenvector centrality* que tiveram um bom desempenho.

No estudo realizado em categoria natural por Rosh et al. [19] , área que estuda a categorização da nomenclatura de objetos, foi constatado que as pessoas caracterizam o mundo em termos de objetos básicos, tais como cadeira ou carro, em vez de conceitos mais abstratos, como móveis ou veículos, ou mais específicos, tais como carro esporte ou cadeira de jardim. Com base nesse estudo, Peroni et al. [11] propôs uma medida para sumarização de ontologia chamada *Name Simplicity*, que visa identificar os conceitos com informação rica em sentido lingüístico. Essa medida foi destacada no trabalho de Li et al. em [8] para realizar uma análise comparativa com outras medidas de características estruturais, constatando-se que a mesma foi satisfatória fornecendo resultados até melhores que as outras medidas em algumas ontologias.

Contudo, baseado nos estudos apresentados de diferentes medidas e nas medidas já utilizadas na versão anterior da ferramenta OWLSum [1,2,3], a ferramenta a ser desenvolvida conterá as melhores medidas estudadas e que atuam em diferentes abordagens. As medidas selecionadas serão detalhadas na próxima seção.

4.2. Medidas de Relevância Adotadas

A relevância de um conceito cn em uma ontologia O é medida considerando: os relacionamentos de cn com outros conceitos em O (Centralidade), a simplicidade de escrita dos rótulos nos conceitos de O (Simplicidade do nome), as ocorrências de cn nas ontologias que formaram O (Freqüência) e a proximidade de c_n com a relevância de todos os outros conceitos em O (Proximidade). Em nossa abordagem, as medidas de Centralidade e Simplicidade do nome são utilizadas para captar a importância de um conceito dentro de uma ontologia. A Freqüência é usada quando uma ontologia O é resultante de um processo de integração (*merging*) de ontologias O1,...,On (por exemplo, ontologias que representam os n pontos que compõem um cluster [3]) e captura as ocorrências dos conceitos nas ontologias geradoras de O. A Proximidade é utilizada em conjunto com a relevância já definida no conceito para fornecer destaque aos conceitos que conecta maior quantidade de conceitos com grande relevância.

4.2.1. Medida de Centralidade

A Centralidade definida em [1,3], que foi baseada na medida *degree centrality* da teoria dos grafos, defende a idéia de que o número de relacionamentos proporciona uma ampla cobertura de acesso entre os conceitos da ontologia e que tipos de relacionamento podem ter pesos diferentes. Os tipos de relacionamentos adotados em [1,3] foram padrões da ontologia, tais como is-a, part-of e same-as; e relacionamentos definidos pelo usuário, com exemplo hasItems ou authorOf. Outro trabalho que utilizou uma variação da medida *degree centrality* e fez avaliação positiva da mesma foi feito por Zhang et al. [6]. A medida *weighted in-degree* usada por Zhang et al. [6], considera o direcionamento das relações para contabilizar o numero de relações de entrada no conceito. Nesse trabalho, juntamos as qualidades das duas medidas de centralidade para aumentar as opções de escolha do usuário. A fórmula de normalização definida para a Centralidade de um conceito cn é:

$$C_I(i) = \sum_{(j,i)\in C} r(j,i) \qquad C_O(i) = \sum_{(i,j)\in C} r(i,j)$$

$$Centrality(c_n) = \frac{(w_I \, x \, C_I \quad + \quad w_o \, x \, C_O) \times \left(\frac{n_s \times w_s}{max_s} + \frac{n_{ud} \times w_{ud}}{max_{ud}} \right)}{|C| - 1}$$

Onde C_I e C_O representam respectivamente o número de conceitos distintos de entrada e de saída, com o qual um conceito cn mantém relacionamento. w_I e w_O são, respectivamente, os pesos de C_I e C_O. ns e nud são, respectivamente, o número de relacionamentos padrão e definidos pelo usuário mantidos por cn. Note que, se cn mantiver mais de um relacionamento com outro conceito, é contabilizado uma única vez. ws e wud são, respectivamente, os pesos dos relacionamentos padrão e definidos pelo usuário. maxs e maxud representam o número máximo de relacionamentos padrão e definidos pelo usuário mantidos por um certo conceito na ontologia. Além disso, (i) *centrality*(cn) $\in$ [0,1]; (ii) ws + wud = 1; (iii) ns + nud = nr; (vi) $w_I + w_O = 1$.

4.2.2. Medida de Simplicidade do nome

A medida *name simplicity* definida em [11] defende a idéia de que categorias naturais normalmente são expressas em rótulos relativamente simples, tais com cadeira e cão. Em outras palavras, considera pouco provável que termos compostos tenham grande valor lingüístico ou relevância para ontologia. Dessa forma é considerado que os conceitos feitos de apenas uma palavra são favorecidos e os compostos de mais de uma palavra são penalizados. Por exemplo, o conceito *Artist* não é penalizado, enquanto que *MusicalArtist* é penalizado, diminuindo o peso da relevância. Baseado nisso foi definida a medida Simplicidade do nome (NS) a seguir:

$$\boldsymbol{NS(C) = 1 - c(nc - 1)}$$

onde *nc* é o número de compostos no rótulo do conceito e *c* é uma constante definida pelo usuário que em [11] foi avaliada contendo o valor c = 0,3. O valor de NS(c) = [0,1]

4.2.3. Medida de Freqüência

A Freqüência é uma medida que pode ser usada em uma ontologia integrada *O*, obtida com o resultado da fusão (*merging*) entre várias ontologias O_1,...,*On* [1,3]. Essa medida foi desenvolvida para avaliar os conceitos de uma ontologia proveniente de uma fusão entre ontologias [3]. A Tabela 1 mostra exemplos de correspondências entre conceitos de *O* e de O_1,...,*On*. Por exemplo, o conceito *Faculty* da ontologia *O* é identificado como: (i) equivalente a *Faculty* na ontologia *O1*; (ii) subconceito de *Worker* na *O2*; e (iii) superconceito de *Professor* em *O3* e *PostDoc* em *O4*.

Tabela 2 - Exemplo de correspondência de conceito

Correspondências para o conceito O:Faculty	
O:Faculty ≡ O1:Faculty	O:Faculty ⊐ O3:Professor
O:Faculty ⊏ O2:Worker	O:Faculty ⊐ O4:PostDoc

Neste trabalho, supomos que *O* possa ser uma fusão ontológica na qual um conceito $c_n \in C$ corresponde a uma ou mais conceitos contidos em O_1, ..., On. Nesse sentido, a fórmula definida para freqüência de um conceito c_n é:

$$frequency(c_n) = \frac{|correspondences(c_n)|}{|O_1,\ldots,O_n|}$$

Em outras palavras, Freqüência é definida como a razão entre o número de conceito correspondentes que envolvam c_n (denotado $|correspondences(c_n)|$) e o número de ontologias diferentes que formaram *O* (indicado por *n*). Ambas as informações podem ser extraídas das correspondências que foram geradas no processo de *merging*. O valor de $frequency(c_n) \in [0,1]$.

4.2.4. Medida de Proximidade

A medida Proximidade foi criada para auxiliar no processo de sumarização, fornecendo destaque aos conceitos que estão próximos dos conceitos de maior relevância. Em outras palavras, a medida do conceito cn é diretamente proporcional ao número de conceitos com grande relevância que estão próximos de cn. A Proximidade não avalia a relevância isoladamente é preciso que os conceitos já tenham uma relevância antes de aplicá-la. O propósito dessa medida é distinguir os conceitos que tenham as mesmas relevâncias, considerando o critério de aproximação a conceitos importantes. Além do mais, como o sumario precisa conter os conceitos mais relevantes, mantendo a integridade relacional, é importante avaliar os conceitos que possam fornecer mais chances de se relacionar conceitos importantes. A fórmula para fornecer um valor ponderado em relação à distância e relevância de um conceito cn é:

$$Closeness(Cn) = \frac{\sum_{n \in C} \frac{relevance(n)}{distance(Cn,n)}}{\sum_{n \in C} 1/distance(Cn,n)}$$

Em outras palavras, Proximidade é uma média ponderada formada pela soma das multiplicações da relevância de n pelo seu peso correspondente, que é representado pela

inversa da distância de n com o conceito cn, divido pela soma dos pesos. Na fórmula *relevance*(n) é a relevância do conceito em n; *distance*(Cn,n) é a menor distância, em número de relações, que consegue interligar os conceitos Cn e n; $\sum n \in C$ é o somatório de cada conceito n que pertence aos conceitos C da ontologia; *Closeness*(Cn) $\in$ [0,1].

5. Algoritmos de Sumarização de Ontologia

Nesse capítulo serão mostrados: o algoritmo utilizado na ferramenta OWLSum, uma análise crítica dos problemas encontrados no algoritmo e uma proposta de novo algoritmo, mostrando melhorias em comparação com o da ferramenta OWLSum.

5.1. Algoritmo definido em OWLSum

Apresentados nos trabalhos de [1,2,3], o algoritmo da ferramenta OWLSum é baseado na idéia de utilizar a relevância do conceito para classificá-lo e, em seguida, gerar um resumo, que tenha o maior número desses conceitos classificados. Em função desse pensamento o algoritmo ficou constituído de 6 passos de processamento.

1. Calcular as relevâncias dos conceitos da ontologia;
2. Classificar os conceitos mais relevantes como RC;
3. Agrupar os conceitos RC que são adjacentes;
4. Identificar os caminhos entre superconceitos e conceitos RC;
5. Analisar a qualidade dos caminhos encontrados;
6. Determinar o melhor caminho com o tamanho esperado.

A Figura 3 demonstra o fluxograma de todas as atividades envolvidas no processo do algoritmo de sumarização. A seguir cada passo será especificado.

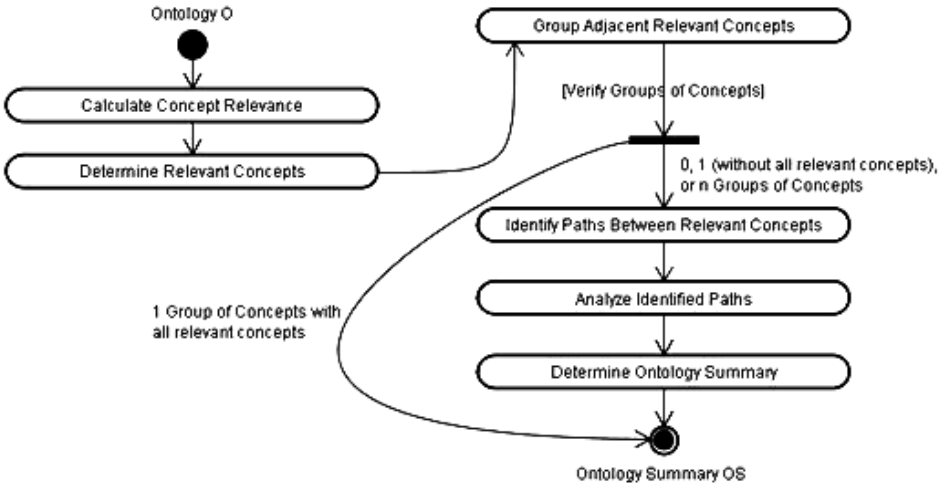

Figura 3 - Fluxograma do algoritmo OWLSum

Passo 1: Calcular as relevâncias dos conceitos da ontologia.

Para calcular a relevância dos conceitos foi utilizada uma função com pesos definidos pelos usuários para fornecer uma proporção dos valores das medidas Centralidade e Freqüência. A fórmula utilizada: relevance(cn) = μ.centrality(cn) + β.frequency(cn). Onde: i) relevance(cn) $\in$ [0,1]; ii) $\mu + \beta = 1$

Passo 2: Classificar os conceitos mais relevantes como RC

Este passo consiste em identificar o conjunto dos conceitos mais relevantes (denominado RC) de uma ontologia O. O algoritmo desenvolvido define a quantidade de classificados através de um modelo híbrido de dois critérios [2]. O modelo seleciona todos os conceitos com relevância igual ou maior à relevância média (RA) da ontologia O para serem candidatos à classificação, e apenas aqueles k conceitos (parâmetro informado pelo usuário) dentro de RA com maior relevância serão considerados conceitos relevantes RC. Segue abaixo a fórmula da relevância média (RA)

$$RA(C) = \sum_{i=1}^{n} \frac{relevance(c_i)}{|C|}$$

Passo 3: Agrupar os conceitos RC que são adjacentes

Este passo consiste em formar grupos de conceitos (superconceitos) que contenham conceitos RC adjacentes da ontologia a ser resumida. Um superconceito deve conter pelo menos 2 conceitos RC adjacentes. Tais agrupamentos são feitos para facilitar o cálculo dos caminhos entre nós relevantes (passo 4). Após a formação dos grupos pode ocorrer uma das cinco situações:

1. Nenhum par de conceitos relevantes adjacentes foi encontrado, portanto nenhum superconceito foi criado.
2. Vários grupos foram encontrados, porém nenhum deles tem tamanho esperado do resumo (informado como parâmetro pelo usuário).
3. Apenas um superconceito de tamanho menor ao esperado foi criado, mais ainda há conceitos RC na ontologia. Nesse caso o processo continua no passo 4 (identificação dos caminhos).
4. Alguns superconceitos com tamanho igual ao esperado pelo usuário foram identificados. Nesse caso, escolhe-se o superconceito de maior relevância média para ser o resumo.

5. Apenas superconceitos com tamanho superior ao tamanho esperado pelo usuário foram identificados. Nesse caso, os superconceitos têm suas folhas de menor relevância eliminadas até ficar no tamanho esperado do resumo e então aquele de maior relevância média é escolhido como resumo.

Passo 4: Identificar os caminhos entre superconceitos e conceitos RC

Este passo consiste em detectar todos os caminhos entre conceitos RC ou superconceitos da ontologia. Cada caminho corresponde a uma subontologia de O, que pode conter conceitos não-relevantes para manter a coerência dos relacionamentos entre os conceitos no caminho.

Passo 5: Analisar a qualidade dos caminhos encontrados

Uma vez que vários caminhos foram identificados, é necessário mensurar a qualidade de cada um através das métricas de cobertura (*recall*) e precisão (*precision*) [2,3]. Essas métricas são utilizadas para determinar o nível de cobertura e precisão de cada caminho (*Path*), respectivamente. *Recall* representa o número máximo de conceitos relevantes no caminho, já *precision* representa o número mínimo de conceitos não-relevantes no caminho.

$$Recall = \frac{|Path_i \cap RC|}{|RC|} \qquad Precision = \frac{|Path_i \cap RC|}{|Path_i|}$$

No algoritmo essas métricas foram utilizadas em conjunto na fórmula *f-measure* [2,3] para combinar precisão e cobertura conforme o parâmetro α, inserida pelo usuário.

$$f - measure = \frac{Precision \times Recall}{(1-\alpha) \times Precision + \alpha \times Recall}, \text{ onde } \alpha \in [0,1].$$

Passo 6: Determinar o melhor caminho com o tamanho esperado

Neste passo a seleção do melhor caminho candidato é feita de acordo com: (i) tamanho que o usuário permitiu através dos parâmetros; (ii) *f-measure*: o caminho deve ter o número máximo de conceitos relevantes e o mínimo de conceitos não-relevante, ou seja, o caminho com o maior valor de *f-measure* deve ser selecionado; (iii) relevância média: caso haja empate no valor de *f-medida* é escolhido o caminho com maior relevância média.

5.2. Análise do Algoritmo OWLSum

Após a apresentação do algoritmo OWLSum podemos analisar algumas lacunas ou inconsistência no seu desenvolvimento. O algoritmo é definido baseado em: (i) definir a relevância dos conceitos, (ii) classificar os conceitos com maior valor de relevância e (iii) gerar um resumo com o maior número possível de conceitos classificados, mantendo as relações da ontologia original. Em função desse pensamento, e da forma como foi definida a parte da classificação dos conceitos, podemos perceber dois problemas não especificados:

a. Como se pode medir a relevância do conceito sem levar em consideração a proximidade com os conceitos mais relevantes, já que os conceitos relevantes têm mais chance de aparecer no resumo? Neste caso, os diretamente ligados a eles também têm maior chance, pois o sumário precisa manter as relações entre os termos.
b. Se o usuário pode delimitar o número de conceitos RC na ontologia, o que acontece com os conceitos que não são RC, por causa da quantidade delimitada de RC que já foi ocupada, mas têm o mesmo valor de relevância de um conceito classificado RC?

Para exemplificar esses problemas apresentamos a ontologia *O* ilustrada na Figura 4, que possui uma rotulagem definida em letras e a relevância representada nas tonalidades em cinza de cada conceito. Na ontologia *O* podemos observar que há conceitos com valores iguais de relevância que ficam ordenados em: {A,C,P}{D,E}{F,G}{B,Q,R,S,V,U,T,X}.

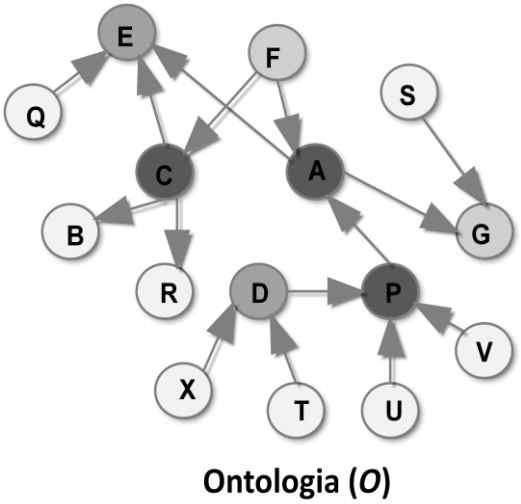

Figura 4 - Ontologia com relevância calculada

Ao colocar os conceitos em seqüência, pudemos perceber que os conceitos formam grupos de mesmo valor de relevância. Em função disso, qual a solução do algoritmo OWLSum para classificar os melhores conceitos de RC, se existir entre os melhores conceitos valores iguais de relevância? O algoritmo não aborda esse cenário. A ordenação dos conceitos é baseada nos valores de relevância, mas como diferenciar valores de relevâncias equivalentes? O algoritmo por não especificar a forma de ordenar os conceitos subentende-se que os conceitos de mesmo valor seguem uma seqüência alfabética.

Problemas como esses acontecem, por exemplo, quando um usuário solicitar um resumo com 4 conceitos relevantes da ontologia *O*. No processo, o algoritmo classifica como RC os conceitos A, C, P e D pelos valores da relevância constatados. Entre esses conceitos classificados, o conceito D apresenta relevância equivalente ao conceito E o qual não apresentou diferença nas medidas de centralidade e freqüência e, no entanto, não foi classificado. A definição de qual o conceito tem mais merecimento de ser classificado como RC entre D e E, é um questionamento que fica claro ao verificar as relações dos mesmos, pois D possui 3 relações sendo uma para um conceito mais relevante e as outras duas para menos relevantes e E possui 3 relações uma para um conceito menos relevante e as outras duas para conceitos mais relevantes. Com isso podemos dizer que as relações de E e de D surtem efeito na relevância do conceito. Para provar que o conceito E tem mais relevância do que o conceito D podemos destacar a resposta da sumarização de *O* como parâmetro de tamanho 4, pois as métricas de *recall* e *precision* nos caminhos {A, P, C, E} e {A, P, D, E} são equivalentes, com 3 conceitos classificados como RC e 1 não classificado, deixando a média de relevância definir o caminho do resumo. Considerando os caminhos, constatamos que o caminho que não contém D tem maior relevância média, portanto em termos da facilidade de agregação com conceitos importantes o conceito E tem mais valor que o conceito D.

5.3. Novo Algoritmo Broaden Paths Relevance (BPR)

Nessa seção serão mostrados: a motivação para a criação do algoritmo e o desenvolvimento do algoritmo BPR, detalhando cada etapa de processamento.

5.3.1. Motivação do algoritmo

Analisando os pontos fortes e fracos do algoritmo OWLSum pudemos observar dois aspectos:

a. Para um resumo de ontologia o mais importante não é ter quantidade de conceitos classificados e sim conter conceitos conectados com os maiores valores de relevância possíveis. Isso pode ser observado no exemplo acima que decidiu o caminho do sumário pelo valor da média de relevância.
b. Para definir a relevância do conceito não se pode considerar apenas as medidas de centralidade, número de relacionamento, e freqüência, número de ocorrência no *merging*; mas, também, levar em consideração a qualidade dos relacionamentos, pois, como foi visto, a aproximação de um conceito para outros com as maiores relevâncias, aumenta as chances de participar no resumo.

Com essas observações, propusemos um algoritmo que possa mensurar a aproximação de um conceito em relação aos mais relevantes sem a necessidade de classificar os conceitos de RC, considerando o valor real da relevância.

O pensamento que segue o algoritmo é encontrar o melhor caminho com os conceitos mais relevantes da ontologia, ou seja, identificar os caminhos de conceitos na ontologia com os maiores valores de relevância através da busca e agregação de conceitos adjacentes de maior relevância.

5.3.2. Desenvolvimento do algoritmo

Inspirado no pensamento do algoritmo Breadth-First Search (BFS) [21] o qual realiza busca que se expande em torno de um vértice no grafo, aumentando gradualmente a área de busca. O BFS, também conhecido como busca em largura, inspirou a criação do algoritmo desse trabalho, por meio de aplicações múltiplas de busca em largura nos conceitos de maior relevância, possibilitando os conceitos mais relevantes expandirem suas conexões até formar um caminho com os melhores conceitos.

Criado a partir do algoritmo OWLSum de [3,2,1] e da influência do BFS, o primeiro passo no processo de sumarização de ontologia, após receber a ontologia e os parâmetros do usuário, é calcular a relevância de cada conceito considerando as medidas de Centralidade, Freqüência, Simplicidade do nome e Proximidade em relação aos parâmetros definidos pelo usuário. Feito isso, precisa se montar uma estrutura para dar continuidade ao processo. A estrutura consiste de três listas: duas voltadas para referenciar conceitos e uma para registrar caminhos candidatos a resumo, ou seja, um conjunto de conceitos interligado que pode ser escolhido como um resumo. A primeira lista NodeSet é igual à lista utilizada no algoritmo anterior do OWLSum que simplesmente ordena os conceitos pelo valor de relevância, sem tratar os conceitos de mesmo valor de relevância. A lista seguinte NodeIntegrity é utilizada para referenciar os conceitos que têm conexões com os caminhos da lista SummarySet, podendo ter a conexão de um conceito com um ou mais caminhos em SummarySet. A última lista SummarySet é responsável em ordenar através de métricas os possíveis caminhos que possam representar o melhor resumo da ontologia.

Após montar a estrutura e ordenar, pela relevância, todos os conceitos em NodeSet, o algoritmo entra em uma busca contínua pelo resumo que satisfaça os parâmetros definidos pelo usuário. Os passos seguintes consistem em verificar no SummarySet a existência de um caminho com alto valor, baseada nas métricas de cobertura de relevância e de grau da média de relevância, para ser o resumo. Caso não encontre, o processo verifica se o conceito *Cn* de maior relevância da lista NodeSet está na lista NodeIntegrity. Caso não esteja, o conceito *Cn* é colocado na lista SummarySet e todos os conceitos que têm conexão com *Cn* são colocados na lista NodeIntegrity. Em seguida é verificado novamente a existência de um resumo em SummarySet e, se não existir, verifica se o conceito *Cn* de maior relevância em NodeSet está na lista NodeIntegrity. Caso o conceito *Cn* esteja em NodeIntegrity é escolhido o melhor conceito *Ci*, baseado na métrica relevância média da junção, para integrar com os caminhos que tenham conexão em SummarySet, podendo realizar junção dos caminhos. O processo continua até encontrar o resumo com os maiores valores de relevância que satisfaçam o tamanho especificado pelo usuário.

Resumidamente, os principais passos do processo de sumarização são:

1. Calcular as relevâncias dos conceitos da ontologia
2. Montar a estrutura do algoritmo e ordenar pela relevância a lista de conceitos
3. Escolher um conceito para integrar aos caminhos candidatos
4. Integrar o conceito escolhido com os caminhos candidatos
5. Verificar a existência de um resumo satisfatório entre os caminhos candidatos

A fim de um melhor entendimento, segue o algoritmo proposto em pseudocódigo no texto abaixo e em fluxograma na Figura 5. A seguir cada passo do algoritmo será detalhado.

Algoritmo Broaden Paths Relevance (BPR)

```
1   Summarizer( Ontology,ParameterUser )
2   NodeSet = Ontology.getConcept
3   CalculateRelevance( NodeSet,ParameterUser )
4   new SummarySet
5   new NodeIntegrity
6   NodeSet.SortRelevance()
7   repeat until ( SummarySet.hasSummary() = false )
8    Concept = NodeSet.getMaxRelevanceConcept()
9    if (NodeIntegrity.hasComparableConcept(Concept) then
10          NodeIntegrity.sort()
11          ConceptIntegrity = NodeIntegrity.getFirstElement()
12          SumarySet.integrate(ConceptIntegrity)
13  else
14          SumarySet.put( Concept )
15          NodeIntegrity.integrateConcepts( Concept.missConected() )
16  end if
17 end repeat
18 return SummarySet.getSummary()
```

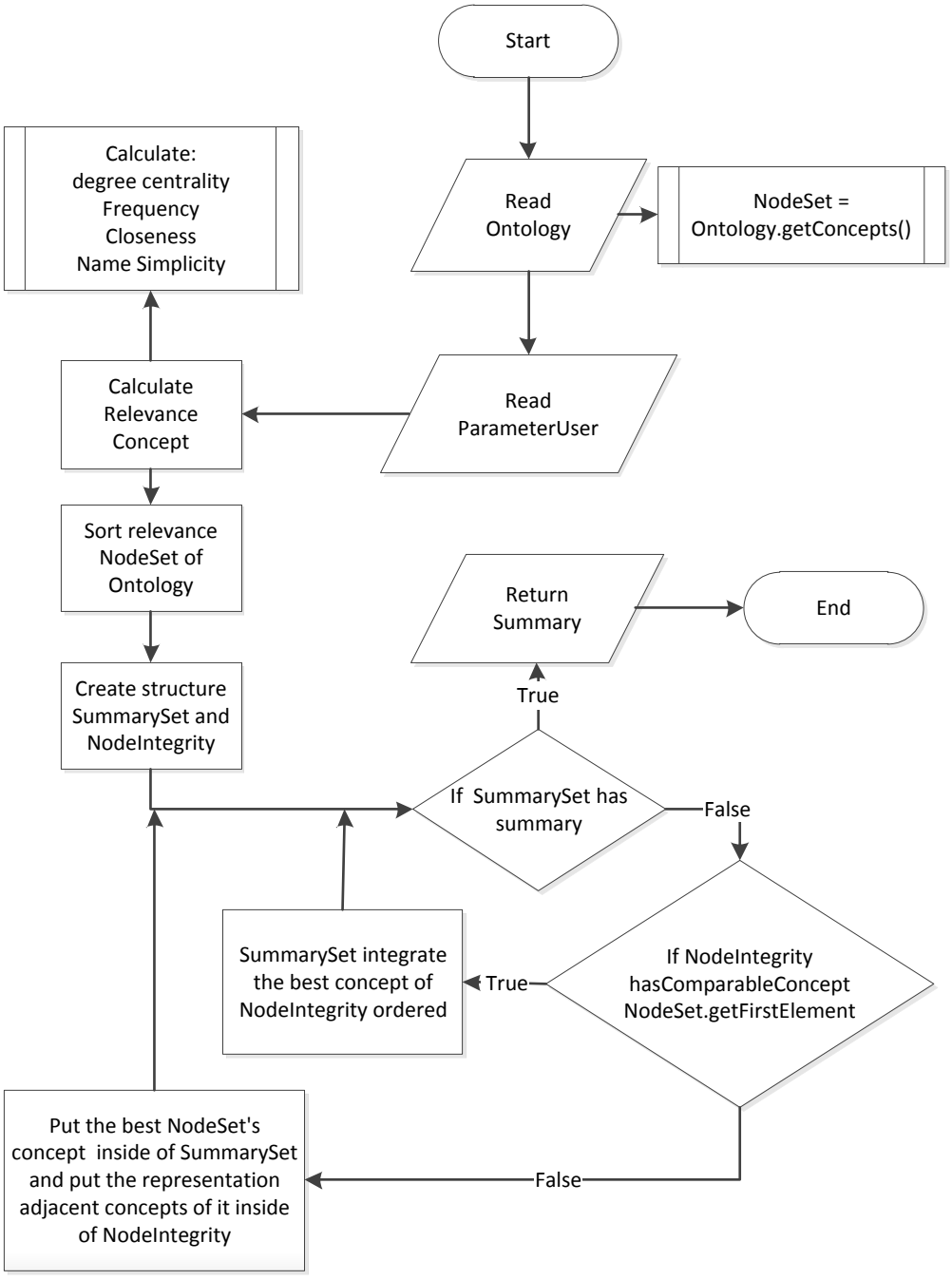

Figura 5 - Fluxograma do algoritmo de sumarização

Passo 1: Calcular as relevâncias dos conceitos da ontologia

Nossa proposta em comparação com a ferramenta OWLSum aborda duas medidas similares, tais como Centralidade e Freqüência que são equivalentes em [1,3], e duas medidas novas Simplicidade do nome encontrada em [11] e Proximidade criada nesse trabalho. Essas medidas já foram bem detalhadas no Capítulo 5 e compõem o cálculo da relevância através de uma fórmula ponderada, com os pesos definidos pelo usuário, de acordo com a importância de cada medida. Segue abaixo a fórmula para cálculo da relevância.

$$relevance(Cn) = \lambda.dCentrality(Cn) + \beta.frequency(Cn) + \gamma.NS(Cn) + \alpha.closeness(Cn)$$

onde *relevance*(c_n) $\in$ [0,1] e os pesos $\lambda + \beta + \gamma + \alpha = 1$.

Passo 2: Montar a estrutura do algoritmo e ordenar, pela relevância, a lista de conceitos

Esta etapa consiste em preparar o ambiente para iniciar a formação do resumo. Começamos ordenando, pela relevância, a lista NodeSet, colocando os mais relevantes no início da lista. Em seqüência criamos duas listas ordenadas NodeIntegrity e SummarySet para ajudar na execução do algoritmo.

A lista NodeIntegrity auxilia o algoritmo referenciando conceitos que têm conexões com os caminhos candidatos da lista SummarySet. A NodeIntegrity foi criada para ordenar os conceitos segundo os valores da relevância de Cn e da soma das relevância dos conceitos dos caminhos dividido pelo número de caminho conectados a Cn. Esses valores seguiram uma proporção baseada no número de conexões de Cn com os caminhos candidatos. Segue abaixo a fórmula relevância das Relações (RR) que calcula essa proporção.

$$RR(C_n) = (1 - \frac{\gamma}{maxR})relevance(C_n) + \frac{\gamma}{maxR}\frac{\sum_{os=1}^{\gamma} relevance(os)}{\gamma}$$

Na fórmula, os parâmetros correspondem a; γ consiste no número de relacionamentos do conceito Cn com caminhos existentes, contado uma única vez um relacionamento de Cn com um caminho; e maxR consiste no maior número de relacionamentos de um conceito que exista na ontologia original. A fórmula RR foi

criada com intuito de influenciar na escolha dos conceitos de mesma relevância em NodeIntegrity, considerando, como indicativo, a relevância de todos os conceitos pertencentes aos caminhos e a quantidade de caminhos que formam conexões com *Cn*.

Já a lista SummarySet é responsável em armazenar os caminhos candidatos a resumo que foram formados pela expansão de conceitos de grande relevância não adjacentes através da busca em largura. A *SummarySet*, também, utiliza métricas para classificar os melhores caminhos e colocá-los em ordem. As métricas definidas para ordenação dos caminhos candidatos foram criadas e nomeadas como cobertura de relevância e grau da média de relevância. A métrica cobertura de relevância (CR) é um cálculo para avaliar a porcentagem de relevância contida no caminho *OS* em relação à ontologia original *O*, formando a seguinte fórmula.

$$CR(OS) = \frac{\sum_{i \subset OS}^{n} relevance(i)}{\sum_{i \subset O}^{n} relevance(i)}$$

Onde ($CR(OS) = [0,1]$), o dividendo é constituído pela soma dos conceitos contidos no caminho *OS* e o divisor é a soma de todos os conceitos contidos na ontologia original *O*. A métrica grau de relevância (DR) avalia o grau de relevância, verificando a relevância média no caminho *OS* em relação à relevância máxima encontrada em um conceito da ontologia *O*. Seguem abaixo as fórmulas da relevância média (RA) e grau de relevância (DR), onde DR(*OS*) = [0,1]

$$RA(OS) = \sum_{i=1}^{n} \frac{relevance(c_i)}{|OS|} \qquad DR(OS) = \frac{RA(OS)}{maxRelevanceConcept(O)}$$

Com a análise dessas duas métricas pudemos classificar os caminhos candidatos através da quantidade de relevância adquirida da ontologia original O e do grau de relevância nos conceitos presentes no caminho *OS*. Nesse caso, utilizamos a fórmula *f-measure* [1,2,3] para combinar DR e CR conforme o parâmetro α, inserido pelo usuário.

$$f - measure(OS) = \frac{DR(OS).CR(OS)}{(1-\alpha).DR(OS)+\alpha.CR(OS)}, \text{ onde } \alpha \in [0,1]$$

A combinação das métricas no *f-measure* é utilizado para classificação e ordenação dos melhores caminhos armazenados em *SummarySet*.

Passo 3: Escolher um conceito para integrar aos caminhos candidatos

Após a preparação do ambiente, inicia-se a busca pelos melhores caminhos que possuem conceitos interligados com os maiores valores de relevância. Este passo consiste em tentar escolher o conceito de maior relevância Cr em duas listas ordenadas por determinados critérios. O processo de escolha consiste em pegar o valor da relevância do primeiro da lista NodeSet, que representa o conceito Cr de maior relevância no momento, para verificar se existe algum conceito diferente com o mesmo valor de relevância na lista NodeIntegrity. Caso não tenha, significa que na lista *SummarySet* não tem nenhum caminho que possa conectar ou ser comparado a Cr e, portanto, o conceito escolhido é o Cr que é o primeiro da lista NodeSet. Caso contrário, existe mais de um conceito representado em NodeIntegrity com valor de Cr, portanto é escolhido o conceito com maior de relevância das Relações na lista NodeIntegrity.

Passo 4: Integrar o conceito escolhido com os caminhos candidatos

Neste passo, após a escolha do conceito Cr, é verificado se existem conexões de Cr com os caminhos contidos na lista *SummarySet*. Caso sejam constatadas conexões entre os mesmos, será realizado um processo de junção de Cr com todos os conceitos dos caminhos em *SummarySet* que são conectados a Cr, formando um caminho maior com todos os conceitos interligados. Em caso de não existir conexões entre Cr e os caminhos em *SummarySet* é criado um novo caminho em *SummarySet* com apenas o conceito Cr. Após a integração de Cr em *SummarySet* através de junção de caminhos ou criando novos caminhos é realizada a inserção em NodeIntegrity de todas as representações dos conceitos que conectam Cr, mas não estão conectados no momento.

Passo 5: Verificar a existência de um resumo satisfatório entre os caminhos candidatos

Este passo consiste em verificar na lista *SummarySet* a existência de algum resumo que tenha o maior valor de *f-measure*, baseado na configuração do usuário, e atenda às especificações definidas pelo usuário na definição dos parâmetros. Caso não tenha um caminho que satisfaça esses requisitos o processo de sumarização de ontologia continua voltando ao passo 3. Caso tenha o caminho que satisfaça esses requisitos o processo acaba e o caminho é retornado como resumo.

5.3.3. Ilustração do algoritmo

Para um entendimento melhor do processo de sumarização da ontologia como um todo será mostrado na Figura 6 o passo-a-passo do processo, visualizando a formação de um resumo de tamanho 4.

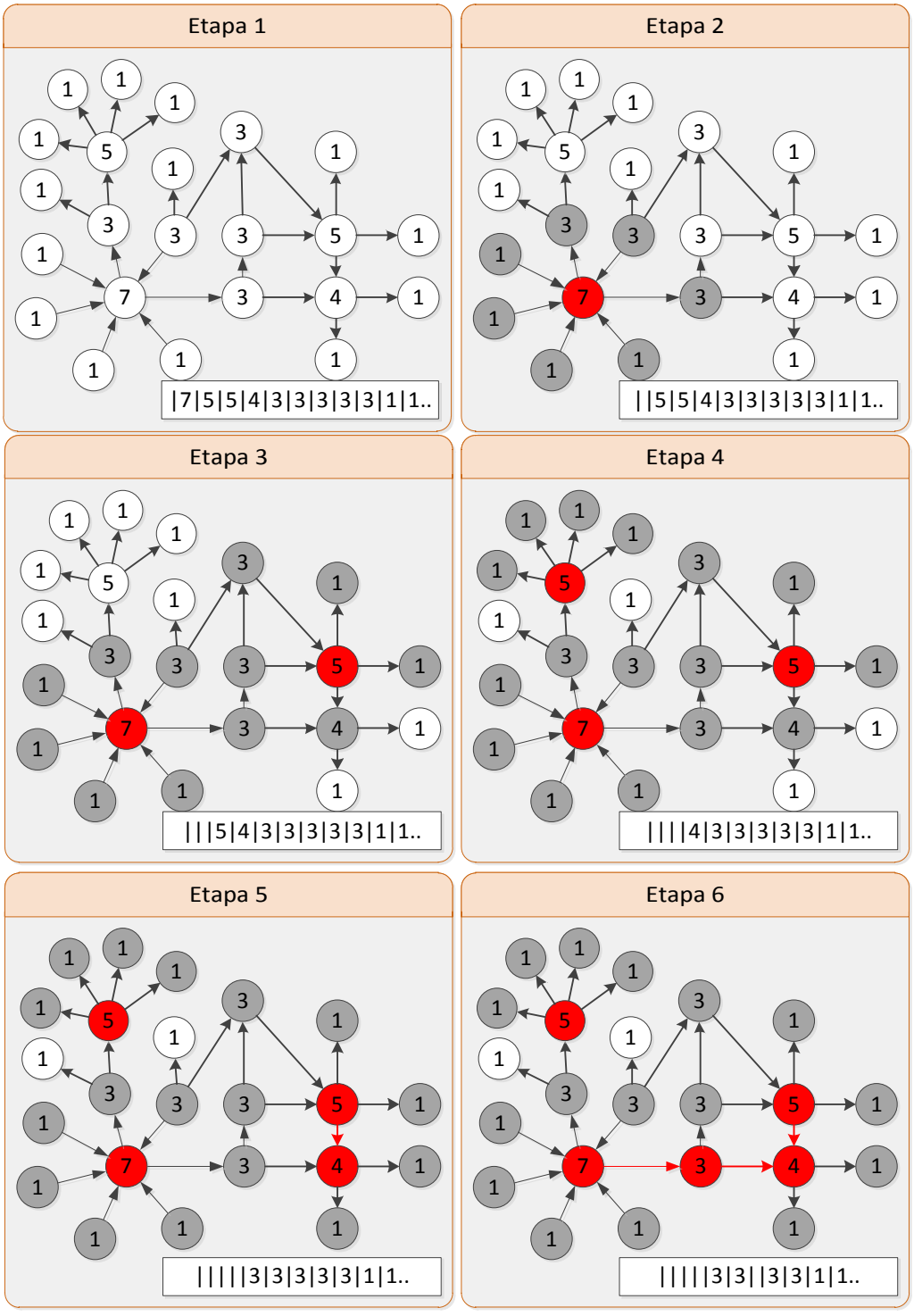

Figura 6 - Processo de sumarização

Para iniciar o processo de sumarização prepara-se o ambiente, ordenando a lista NodeSet, como é apresentado na Figura 6, e criando as listas NodeIntegrity e *SummarySet* que serão representadas, respectivamente, pelas colorações cinza e vermelho com setas vermelhas para a identificação dos caminhos. A etapa 1 da figura mostra a inicialização do processo apresentando a relevância de cada conceito em números e a preparação do ambiente com a ordenação da lista NodeSet. Na etapa 2 é realizada a escolha do conceito de maior relevância disponível na lista NodeSet para ser colocado na lista *SummarySet*, como mostra o conceito 7 em vermelho, e os conceitos que se relacionam com o mesmo, com coloração em cinza, para a lista NodeIntegrity. Na etapa 3 é realizada a escolha do próximo conceito com relevância de valor 5, que não tem conceitos comparáveis em NodeIntegrity, para ser colocado na lista *SummarySet* e os conceitos que fazem conexão com ele, na lista NodeIntegrity. A etapa 4 realiza o mesmo processo da etapa 3. Na etapa 5 o conceito de relevância 4 tem conexão com o conceito que está na lista *SummarySet*, realizando uma junção entre os dois, como mostra a figura através das setas em vermelho, e os conceitos que se relacionam com o mesmo vão para a lista NodeIntegrity. Finalmente, na etapa 6, com três sumários na lista *SummarySet*, é realizado o processo de escolha do próximo conceito que será de relevância com valor 3. Essa escolha será diferente das outras etapas, pois há conceitos de valor 3 comparáveis na lista NodeIntegrity. Portanto, o conceito a ser selecionado terá o maior valor em relação às conexões realizadas com os caminhos da lista *SummarySet*. Nesse cenário o conceito escolhido tem relação com {7} e {5,4} formando um peso equivalente a 16, superando o outro que conecta {7} e {5}, formando um peso de 12. Realizando a verificação após cada inserção na lista *SummarySet*, o processo de sumarização é finalizado encontrando um sumário com o tamanho 4 solicitado, formado por {7,3,4,5}

6. Resultados e Discussão

Para a avaliação das ferramentas utilizaremos a ontologia[1] networkA.owl, que descreve a área de uma rede local, que é a mesma utilizada em [3]. No processo avaliativo das ferramentas realizamos comparações de resultados com base nos parâmetros de entrada de forma que fornecem valores de relevâncias equivalentes para os conceitos da ontologia nas duas ferramentas. Em função disso, definimos que apenas a medida Centralidade seria utilizada no calculo da relevância, pois entre as duas medidas presentes nas ferramentas ela é a única que pode ser aplicada nesta ontologia, devido à restrição da medida Freqüência às ontologias resultantes do processo de *merging*. Nesses experimentos observamos valores de relevância equivalentes nas duas ferramentas devido à aplicação da mesma medida. Como mostra na Tabela 3

Tabela 3 - Relevância dos Conceitos pela medida Centralidade

Conceitos	Relevância
ServerSoftware	0.23076923076923078
Software	0.19230769230769232
SwitchEquipment	0.19230769230769232
Computer	0.19230769230769232
NetworkNode	0.19230769230769232
Cable	0.19230769230769232
SecurityEquipment, Equipment	0.11538461538461539
NodePair, RoutingComputer	0.07692307692307693
OfficeSoftware, FTPServer, Switch, HardwareSniffer, WebServer, OperatingSystem, SSHServer, OtherServer, CrossOverCable, StraightThroughCable, Hub, CoaxCable, Server, HardwareFirewall, PC, Router, TelnetServer	0.038461538461538464

[1]http://www.atl.lmco.com/projects/ontology/ontologies/network/networkA.owl

Através dessas relevâncias podemos destacar níveis de valores entre os conceitos, como está ilustrada na Figura 7.

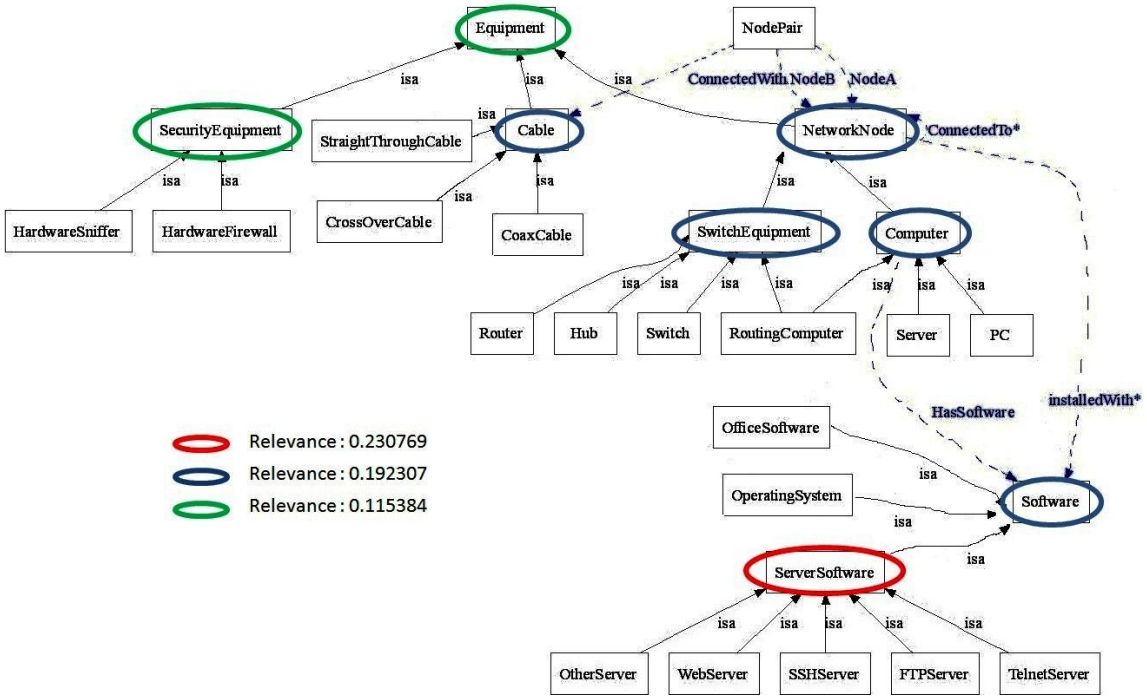

Figura 7 - Relevantes conceitos da ontologia networkA.owl

Nessas configurações foi solicitado para as duas ferramentas um resumo de tamanho 4, não permitindo qualquer variação do tamanho. Para avaliarmos os resultados consideramos os indicativos que definem uma ótima resposta das duas ferramentas. Na OWLSum o melhor resumo é considerado quando possui apenas conceitos classificados em RC, já para a ferramenta OWLSumBPR é contendo os conceitos de maiores relevâncias. Com base nesses indicativos os resumos foram limitados a terem apenas os conceitos RC em OWLSum e os conceitos mais relevantes em OWLSumBPR, respectivamente. Na ontologia networkA.owl essa limitação não é prejudicial na formação de um resumo, pois como foi observado na Figura 7 os 5 conceitos mais relevantes são adjacentes. Seguindo essas exigências, realizamos os experimentos nas ferramentas e constatamos tais resultados.

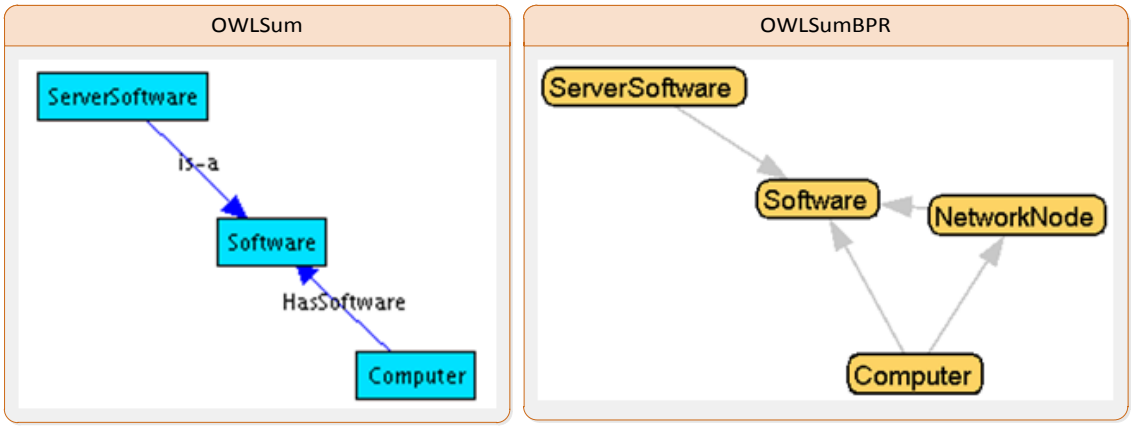

Figura 8 – Comparação de resultados entre as ferramentas OWLSum e OWLSumBPR

A ferramenta OWLSum gerou um resumo com 3 conceitos, os quais são todos classificados de RC e a ferramenta OWLSumBPR criou um resumo com os 4 conceitos mais relevantes. A fim de tentar entender o motivo da ferramenta OWLSum ter gerado um resumo com 3 conceitos, realizamos outro experimento possibilitando a variação do tamanho. Na nova execução foi gerado uma ontologia com 5 conceitos tendo 4 conceitos classificados em RC. Como pode ser visto na Figura 9.

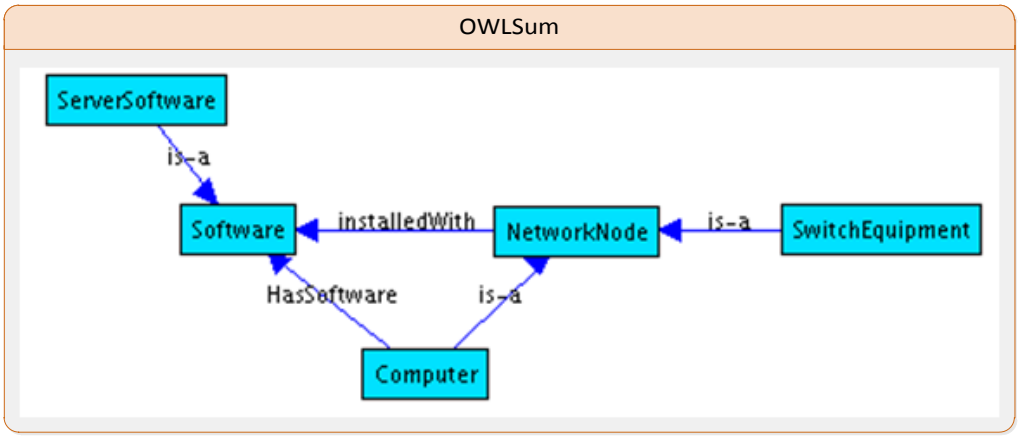

Figura 9 - Resumo com tamanho 4 com variação

Esse resultado irregular foi gerado devido à ineficácia da classificação dos conceitos, pois a importância da classificação do conceito predomina entre os conceitos, mesmo se tiverem valores de relevância equivalentes. Esse ato de classificar os conceitos prejudica a formação de um resumo, pois não considera os relacionamentos e a relevância do conceito de forma combinada. Um exemplo claro disso são os resultados mostrados nas Figuras 8 e 9, que devido às classificações terem seguido a ordem dos valores de relevância, não realizaram uma classificação adequada para formação do

resumo. Nos resumos ilustrados das Figuras as classificações dos conceitos seguiram a ordem mostra na Tabela 4.

Tabela 4 - Classificação dos conceitos de networkA.owl

Conceitos	Classificado RC	Relevância
ServerSoftware	X	0.23076923076923078
Software	X	0.19230769230769232
SwitchEquipment	X	0.19230769230769232
Computer	X	0.19230769230769232
NetworkNode		0.19230769230769232
Cable		0.19230769230769232

Como podemos perceber na Tabela 4 os conceitos {Software, SwitchEquipment e Computer} foram classificados como RC e os conceitos { NetworkNode e Cable}, de mesmo valor de relevância, não foram classificados. Em função disso os conceitos { ServerSoftware, Software, e Computer} mostrados na Figura 8 não puderam se relacionar com o conceito SwitchEquipment, tendo, então, que se conformar com a geração de um resumo de tamanho 3 ou de tamanho 5 se for permitido inserir um conceito não classificado NetworkNode, que possibilite a conexão entre os conceitos. Como vimos os valores de SwitchEquipment e NetworkNode são iguais e a ferramenta OWLSum poderia gerar um resumo igual a OWLSumBPR { ServerSoftware, Software, Computer e NetworkNode } se tivesse classificado o conceito NetworkNode no lugar do SwitchEquipment. Portanto, classificar um conceito sem considerar os relacionamentos, que auxiliam e definem a integração dos conceitos mais relevantes, é um erro que afeta o processo de sumarização como um todo. Por isso, para a realização adequada de um resumo precisam-se combinar os valores das relevâncias com as relações dos conceitos.

7. Aspectos da implementação

Neste capítulo serão apresentados alguns detalhes importantes da implementação, bem como instruções de como utilizar a ferramenta. Alguns experimentos serão descritos e validados a fim de demonstrar a coerência entre o que foi projetado (capítulo 3) e o que foi implementado.

7.1. Ferramenta desenvolvida

A ferramenta implementada foi denominada OWLSumBPR, uma vez que é a evolução da ferramenta OWLSum com a aplicação de um novo algoritmo denominado *Broaden Paths RelevanceI* (BPR). A ferramenta, implementada em Java, pode ser dividida em dois componentes: o da interface, para visualização da ontologia em formato de grafo, e o de processamento de sumário, para realização do processo de sumarização de ontologia. O componente de visualização utilizou uma interface open source *owl2prefuse* [9] para visualizar em formato de grafo os arquivos OWL. Já o componente de processamento de sumarização utilizou a API Jena para manipular a ontologia e permitir a aplicação do algoritmo. O propósito da ferramenta é poder ser ofertada como uma API, sendo disponibilizada e facilmente incorporada em aplicações Java.

7.2. Componente de visualização

A Figura 10 representa o componente de visualização que recebe um arquivo OWL e gera uma interface gráfica de um grafo. O componente foi alterado para ilustrar uma ontologia com os conceitos provenientes do processo de sumarização, destacados na cor rosa como mostra a Figura 10.

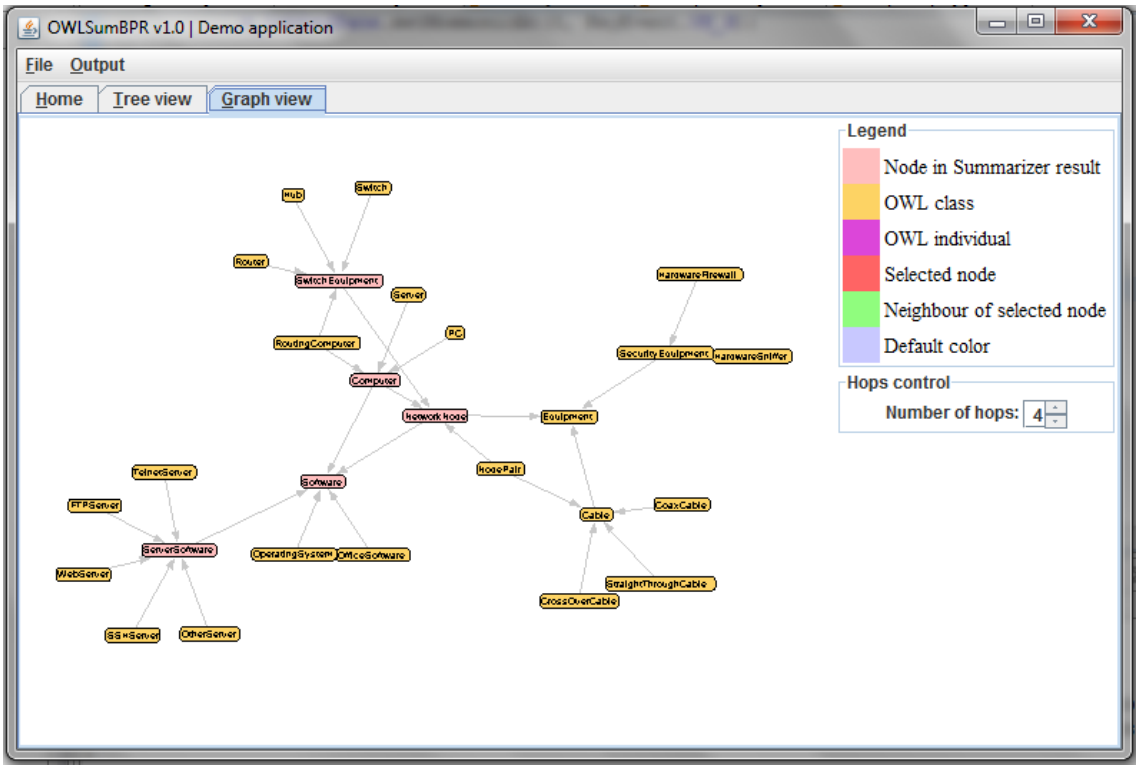

Figura 10 - Componente de visualização da OWLSumBPR

7.3. Componente de processamento de sumário

A parte de processamento de sumarização da API consiste de um módulo que recebe os parâmetros de configuração para realizar a sumarização. Tais parâmetros são:

i. Nome do arquivo OWL de entrada a ser resumido;
ii. Nome do arquivo OWL de saída que conterá o resumo;
iii. Nome do arquivo XML contendo os mapeamentos ontológicos;
iv. Tamanho desejado do resumo;
v. Valor entre 0 e 1 do peso da centralidade no cálculo da relevância;
vi. Valor entre 0 e 1 do peso da freqüência no cálculo da relevância;
vii. Valor entre 0 e 1 do peso da proximidade no cálculo da relevância;
viii. Valor entre 0 e 1 do peso da simplicidade do nome no cálculo da relevância;
ix. Valor entre 0 e 1 do alfa da média harmônica *f-measure* para cálculo da qualidade dos caminhos;
x. Valor entre 0 e 1 do peso para o tipo de relacionamento no cálculo da medida centralidade;

Apenas os parâmetros (i), (ii) e (iv) são obrigatórios na inserção dos parâmetros. Uma vez recebidos os parâmetros, a ferramenta converte a ontologia informada em um grafo de forma que os conceitos viram vértices e os relacionamentos arestas. Na conversão em grafo, arestas relativas a auto-relacionamentos (quando origem e destino da aresta correspondem ao mesmo nó) são desconsideradas no cálculo do resumo assim como as arestas múltiplas que têm vários relacionamentos de um nó para outro, considerando apenas uma única aresta. Ao final, depois que os cálculos da relevância e o resumo foram realizados, as arestas múltiplas e os auto-relacionamentos voltam a aparecer no grafo e então o grafo é convertido na ontologia OWL resumida.

8. Conclusão

Este trabalho apresentou a ferramenta OWLSumBPR, uma evolução da OWLSum, que soluciona algumas falhas e apresenta novas configurações. A ferramenta OWLSumBPR desenvolvida, conta com um novo processo para sumarização de ontologia, utilizando o algoritmo *Broaden Paths Relevance* (BPR), que excluiu do processo anterior a classificação dos conceitos para formar resumos através da expansão de caminho de conceitos que são relevantes e conectados.

A nova ferramenta manteve as funcionalidades da anterior OWLSum e ampliou as possibilidades de configuração com novas medidas de relevância. A OWLSumBPR manteve a facilidade de portabilidade em aplicativos Java e dispõe das medidas de: aproximação de relevância, grau de centralidade, freqüência e nome simplificado. Com essas melhorias há um aumento nas possibilidades de configurações, permitindo ao usuário encontrar diferentes tipos de resumo em função dos parâmetros de tamanho e das variações de medidas definidas.

Este documento, em conjunto com os dois experimentos demonstrados, evidencia a falha da ferramenta OWLSum, mostrando um solução com a ferramenta OWLSumBPR sem perder qualidade de formação dos resumos customizáveis.

8.1. Trabalhos futuros

Durante o desenvolvimento da ferramenta OWLSumBPR observou-se que o algoritmo BPR pode ampliar as formas de solicitar a sumarização de uma ontologia, possibilitando realizações de sumários em função de determinadas condições, como por exemplo:

a. Formar um resumo da ontologia em torno de um determinado conceito, realizando uma expansão no caminho que contenha o conceito.
b. Formar um resumo da ontologia dos conceitos que são subclasses de um conceito determinado, ou seja, restringir o resumo apenas para os conceitos que são subclasses de um conceito definido pelo usuário.

Essas novas possibilidades de sumarização podem ser úteis como componentes de diferentes aplicações, permitindo expandir as formas de resumir uma ontologia. Além desses trabalhos, a ferramenta poderia melhorar seus componentes para disponibilizá-los na Web como uma API para sumarização de ontologia.

9. Referências

[1] P. O. Sousa, C. E. Pires, and A. C. Salgado, "Uma Ferramenta de Sumarização de Esquemas Representados por Ontologias," in *(SBBD'10) Simpósio Brasileiro de Banco de Dados*, Belo Horizonte, Brasil, 2010.

[2] V. B. Alencar, "Uma Ferramenta para Sumarização de Ontologias", 2008, Universidade Federal de Pernambuco (UFPE/CIn). Monografia de Conclusão da Graduação. Recife, PE, Brasil.

[3] C. E. Pires, "Ontology-based Clustering in Peer Data Management System", 2009, Universidade Federal de Pernambuco (UFPE/CIn). Tese de Doutorado, Recife, PE, Brasil. pp. 89-105.

[4] B. Smith, "Ontology (Science)," in *C. Eschenbach and M. Gruninger (eds.), Formal Ontology in Information Systems. Proceedings of FOIS 2008*, Amsterdam/New York: ISO Press, 2008, pp. 21-35.

[5] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, pp. 199-220, April 1993.

[6] X. Zhang, G. Cheng, and Y. Qu, "Ontology Summarization Based on RDF Sentence Graph.," in *(WWW'07) The 16th International World Wide Web Conference*, Banff, Alberta, Canada, 2007, pp. 707-715.

[7] S. Staab, D. Oberle, and N. Guarino, "What Is an Ontology?," in *Handbook on Ontologies*, 2nd ed., S. Staab and R. Studer, Eds. Berlin, Germany: Springer Verlag, 2009, pp. 1-20.

[8] N. Li, E. Motta, and M. d'Aquin, "Ontology summarization: an analysis and an evaluation.," in *(IWEST'10) The International Workshop on Evaluation of Semantic Technologies*, Shanghai, China, 2010.

[9] J. Borsje and J. Giles. (2011, Dezembro) OWL2Prefuse project. [Online]. http://owl2prefuse.sourceforge.net/

[10] N. Li and E. Motta, "Evaluations of user-driven ontology summarization.," in *(EKAW'10) The 17th International Conference on Knowledge Engineering and Knowledge Management by the masses*, Lisbon, Portugal, 2010, pp. 11-15.

[11] S. Peroni, E. Motta, and M. d'Aquin, "Identifying Key Concepts in an Ontology Through the integration of cognitive principles with statistical and topological

measures," in *(ASWC'08) The 3rd Asian Semantic Web Conference*, Bangkok, Thailand, 2008.

[12] K. Okamoto and W. and Li, X. Chen. Ranking of Closeness Centrality for Large-Scale Social Networks.

[13] D. Das and A. F.T. Martins, "A Survey on Automatic Text Summarization," *Literature Survey for the Language and Statistics II Course at CMU*, vol. 4, pp. 192-195, 2007.

[14] H. Stuckenschmidt and M. Klein, "Structure-Based Partitioning of Large Concept Hierarchies," in *(ISWC'04) International Semantic Web Conference*, Hiroshima, Japan, 2004.

[15] M. D'Aquin, A. Schlicht, H. Stuckenschmidt, and M. Sabou, "Criteria and Evaluation for Ontology Modularization Techniques," in *Modular Ontologies*, S. Heiner, C. Parent, and S. Stefano, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 67-89.

[16] M. Sabou, V. Lopez, E. Motta, and V. Uren, "Ontology Selection: Ontology Evaluation on the Real Semantic Web," in *(WWW'06) Workshop: Evaluation of Ontologies for the Web at 15th International World Wide Web Conference*, Edinburgh, Scotland, 2006, pp. 23-26.

[17] M. K. Smith, C. Welty, and D. L. McGuinness. (2010, Dezembro) OWL Web Ontology Language Guide: W3C Recommendation 10 February 2004. [Online]. http://www.w3.org/TR/owl-guide/

[18] I. Horrocks, F. V. Harmelen, and P. Patel-Schneider. (2010, Dezembro) DAML+OIL (March 2001). [Online]. http://www.daml.org/2001/03/daml+oil-index

[19] E. Rosch, C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyes-braem, "Basic objects in natural categories," *Cognitive Psychology*, vol. 8, no. 3, pp. 382-439, July 1976.

[20] M. E. J. Newman, *Networks: An Introduction*.: Oxford University Press, 2010.

[21] S. K. Das and C. C.-Y. Chen, "A new parallel algorithm for breadth-first search on interval graphs," in *Parallel Processing Symposium*, Beverly Hills, CA , USA , 1992, pp. 150 - 153.

[22] C., Millard, D. and Howard, Y. Kousetti. (2008) A Study of Ontology Convergence in a Semantic Wiki. In: WikiSym 2008, September 8-10, Porto, Portugal.