UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

ROGER BATTASSINI

Uma ferramenta para busca temporal na DBPedia a partir de uma ontologia

Trabalho de Graduação.

Profª. Drª. Renata Galante
Orientadora

Leandro Gallina
Co-orientador

Porto Alegre, dezembro de 2011.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface* (em português, Interface de Programação de Aplicativos)
CSV	*Comma-separated values* (em português, Valores Separados por Vírgula)
DTD	*Document Type Definition* (em português, Definição de Tipo de Documento)
HTML	*HyperText Markup Language* (em português, Linguagem de Marcação de Hipertexto)
HTTP	*Hypertext Transfer Protocol* (em português, Protocolo de Transferência de Hipertexto)
JRE	*Java Runtime Environment* (em português, Ambiente de Tempo de Execução Java)
JSON	*Javascript Object Notation* (em português, Notação de Objeto Javascript)
OWL	*Web Ontology Language* (em português, Linguagem de ontologia Web)
PHP	*Hypertext Processor* (em português, Processador de Hipertexto)
QA	*Question Answering* (em português, busca de respostas)
RDF	*Resource Description Framework* (em português, arcabouço de descrição de recurso)
SPARQL	*SPARQL Protocol and RDF Query Language* (em português, Protocolo SPARQL e Linguagem de Consultas RDF)
SQL	Structured Query Language (em português, Linguagem de Consulta Estruturada)
URI	*Uniform Resource Identifier* (em português, Identificador Uniforme de Recurso)
URL	*Uniform Resource Locator* (em português, Localizador de Recurso Uniforme)
US	*United States* (em português, Estados Unidos)
XML	*Extensible Markup Language* (em português, Linguagem Padronizada de Marcação Genérica)
XPath	*XML Path Language* (em português, Linguagem de Caminho XML)

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Este trabalho apresenta uma ferramenta Web destinada a responder perguntas com relevância temporal que são feitas em linguagem natural sobre dados da DBPedia. Os dados pesquisados foram previamente capturados de páginas da Web e armazenados em um formato estruturado seguindo as regras definidas em uma ontologia.

A principal contribuição deste trabalho é propor e implementar uma ferramenta capaz de responder perguntas temporais em linguagem natural em inglês através de consultas bem-definidas sobre um conjunto de dados estruturados obtido a partir de páginas da Web seguindo as regras da ontologia da DBPedia. Essa resposta é dada também na linguagem do usuário.

Os experimentos realizados mostram uma alta precisão nas respostas fornecidas pela ferramenta, o que indica que esse tipo de representação estruturada de dados temporais proporciona resultados de boa qualidade para as buscas realizadas.

Palavras-Chave: consulta temporal, busca de respostas, DBPedia, ontologia.

ABSTRACT

This document presents a Web tool which aims to answer questions with temporal relevance made in natural language over DBPedia data. The searched data have been previously retrieved from Web pages and stored in a structured format following the rules defined by an ontology.

The experiments made show a high precision on the answers given by the tool, which indicates that this structured temporal data representation grants good quality results for the searches done.

The main contribution of this work is proposing and implementing a tool capable of answering natural language temporal questions in english through well-defined queries over a structured dataset obtained from Web pages following the rules of the DBPedia ontology. The answer is given in the user's language aswell.

Keywords: temporal queries, question answering, DBPedia, ontology.

1 INTRODUÇÃO

O número de páginas disponíveis para acesso na Internet vem crescendo a uma grande velocidade, e os mecanismos de busca por palavras-chave nessas páginas têm se tornado um serviço essencial. Fazem-se necessárias ferramentas avançadas para que se encontre o que é buscado com mais eficiência e rapidez.

Uma importante característica desses documentos é a informação temporal, ou seja, referências a termos que, diretamente ou não, representam uma entidade de tempo. Por exemplo, "***Semana passada*** *choveu o esperado para todo o mês, (...)*" ou "***No dia da Proclamação da República****, a imprudência dos motoristas fez mais uma vítima em São Paulo.*". Poucos são os casos em que essa informação aparece em um formato absoluto e normalizado, como em "***No dia 3 de março de 2008, às 13:30****, houve um terremoto na Malásia.*". O texto normalmente é redigido em linguagem natural, podendo haver até mesmo gírias ou termos locais, e os autores podem ser basicamente qualquer pessoa com acesso à rede, portanto identificar essa informação pode não ser tão simples.

Muitos eventos que aparecem nos documentos têm sua semântica condicionada à época aos quais se referem, enquanto outros se repetem com uma determinada frequência, de modo que ignorar os termos temporais contidos no próprio texto pode significar não fornecer ao usuário o que ele pretende visualizar, ou até mesmo retornar uma informação desatualizada ou incorreta.

As ferramentas de busca atuais basicamente têm seu conhecimento temporal limitado à data de publicação das páginas e possivelmente algumas datas fixas que coincidem com as palavras-chave submetidas pelo usuário, o que está se mostrando insuficiente. Ao passo que a quantidade de informações e a linha de tempo disponíveis para pesquisa crescem, também aumenta o risco dessas ferramentas indicarem páginas antigas a alguém que pesquisou especificamente algum evento atual.

O objetivo desse trabalho é especificar e implementar uma ferramenta de busca temporal na DBPedia ou em uma fonte de informações que siga as regras definidas pela mesma ontologia utilizada pela DBPedia. Essa fonte de informações deve conter relações específicas sobre entidades e informações temporais. Essa informação formatada é anteriormente obtida a partir da detecção, extração e normalização de informações temporais em um conjunto de documentos Web.

A ferramenta desenvolvida neste trabalho se baseia em pesquisa através de perguntas em linguagem natural na língua inglesa, simulando algo próximo do que uma pessoa perguntaria a outra. São suportadas várias modalidades de perguntas, que são automaticamente detectadas analisando as palavras que compõem a pergunta, e variam dependendo do que o usuário quer saber e o que ele fornece na própria pergunta. São feitas otimizações para garantir: (i) abrangência, isto é, caso exista apenas uma

informação no conjunto de dados que responde parcialmente à pergunta, ela é retornada; e (ii) precisão, ou seja, se houver diversas entradas que podem responder à pergunta, aquela que está mais próxima ou coincide com o que foi pesquisado é a escolhida para aparecer primeiro. As respostas são dadas em linguagem natural também, fazendo com que o mecanismo de busca tenha um comportamento semelhante a um diálogo, o que certamente torna a aplicação mais amigável para o usuário. A ferramenta é feita para a Web, sendo utilizada através de um navegador e atuando sobre dados originalmente extraídos de documentos da Web.

Os experimentos realizados mostraram que a ferramenta tem uma alta precisão nos resultados obtidos em responder perguntas com relevância temporal sobre o conjunto de dados definido seguindo as regras da ontologia da DBPedia, comparando-se com os resultados esperados.

A organização do trabalho é como segue: o capítulo 2 aborda a fundamentação teórica necessária para a compreensão do trabalho, com definições de assuntos importantes para o trabalho e a descrição das ferramentas utilizadas para o desenvolvimento da ferramenta. O capítulo 3 se destina a falar sobre trabalhos relacionados a este. O capítulo 4 apresenta a ferramenta proposta com detalhes. No capítulo 5, são descritos os experimentos realizados sobre a ferramenta desenvolvida e são discutidos os resultados obtidos. Por fim, a seção 6 é uma conclusão de todo o trabalho desenvolvido.

2 FUNDAMENTAÇÃO TEÓRICA

O objetivo desse capítulo é familiarizar o leitor com os principais conceitos e ferramentas utilizados na especificação e implementação da ferramenta proposta. A primeira seção define informação temporal, quais os termos que a definem, como e de que forma costumam aparecer, os desafios envolvidos em detectar e normalizar esses dados, as razões pelas quais os motores de busca atuais têm problemas em explorá-los devidamente, e qual a importância de levar-se em conta esse tipo de informação. A seção 2.2 aborda os dados interligados, definindo conceitos como ontologias, o modelo para representação de dados interligados, RDF, e a linguagem para consulta sobre esses dados, SPARQL. A seção 2.3 descreve as ferramentas utilizadas no trabalho e a seção 2.4 encerra o capítulo com as considerações finais sobre o que foi abordado nessas seções.

2.1 Informação Temporal

A informação temporal, como o nome sugere, engloba quaisquer termos que fazem referência a, ou podem ser convertidos em entidades de tempo, como, por exemplo, datas, durações, épocas ou outros eventos com valor temporal. Os exemplos citados de agora em diante estão grafados no idioma inglês, para estar de acordo com o trabalho realizado sobre uma ontologia nessa língua. Isso se deve ao fato da grande maioria das informações da Internet estar em inglês. Conforme estudo feito em (UNIÃO LATINA, 2007), no ano de 2007, 45% das páginas da Web estavam escritas em inglês, contra apenas 1,39% em português. A ontologia da DBPedia também está neste idioma, e a implementação de um sistema QA é bem mais simples devido à simplicidade da sintaxe do inglês.

Segundo (SNODGRASS; AHN, 1985), uma data é dita absoluta quando seu conteúdo está explícito no texto. Por exemplo, o trecho "*Bill Gates founded Microsoft in 1975*" indica uma data absoluta, porém incompleta, pois têm-se apenas o ano. Já no trecho "*Bill Clinton was the president between 1993 and 2001*" está definida uma duração de tempo absoluta. O dado temporal é um período desde 1993 até 2001. Por fim, o trecho "*Steve Jobs was born on 02/24/1955*" indica uma data absoluta e completa, pois têm-se o dia, mês e ano.

(SNODGRASS; AHN, 1985) define os eventos temporais relativos como aqueles que contêm um termo com valor temporal, mas não o termo diretamente. "*Pluto stopped being considered a planet eight years ago*" e "*That building was completed during Carnival*" são exemplos disso, pois apesar de haver uma informação com valor temporal semanticamente associada aos eventos descritos, a data ou período exato não são trivialmente conhecidos na frase, e precisam ser inferidos. No primeiro caso, o método de inferência seria mais simples, apenas tendo-se que deslocar o ano atual. Já no

segundo, deve haver uma forma de descobrir a qual dado temporal absoluto o Carnaval está associado.

Existem inúmeras maneiras de ligar um sujeito e um predicado a um tempo. Vários autores utilizam técnicas de emprego de sinônimos e paráfrases para evitar repetições no seu texto, e, além disso, existem termos locais, gírias e mais formas de se expressar uma mesma informação em linguagem natural. Isso dá uma ideia da dinâmica envolvida em se identificar tudo que faz alusão ao tempo em um texto qualquer.

Para fins deste trabalho, define-se que uma informação temporal é um elemento do texto, ligado a um evento e a um sujeito que praticou esse evento, a partir do qual pode ser extraída:

a) uma data (completa ou não);

b) um intervalo entre duas datas do item (a);

c) outro evento, a partir do qual obtém-se um dos itens anteriores

Para um humano atualizado no assunto da pergunta, é fácil responder a questões como "Is Barack Obama the US president now?" ou "What happened in Rio de Janeiro last week?". Porém, os motores de busca tradicionais costumam ter problemas para responder perguntas desse tipo, priorizando mais as páginas que são frequentemente acessadas e que contenham o máximo possível de palavras-chave coincidentes, ponderando esses dois fatores. Isto significa que o tempo relativo, na maioria das vezes, não é inferido pelos motores de busca.

É muito importante identificar quando uma pesquisa está ligada a um trecho específico da linha de tempo, pois só assim é possível distinguir qual a resposta que o usuário está de fato procurando. Em muitos casos, existem várias candidatas a resposta para aquela pergunta, mas só algumas se situam no mesmo período de interesse temporal. Sendo assim, levar em conta a informação temporal presente em um texto e utilizá-la para realizar a busca tem papel vital na precisão dos resultados obtidos nesse tipo de pesquisa.

A informação temporal é utilizada no trabalho porque os dados sobre os quais as consultas são feitas guardam as ocorrências de tempo de páginas Web em um formato padronizado, mas essas informações não estão padronizadas nos documentos originais, sendo necessário inferir datas absolutas sobre datas relativas e normalizar as datas absolutas para um formato compatível com os dados da DBPedia.

2.2 Dados Interligados: Representação e Consulta

O conceito de dados interligados (BERNERS-LEE, 2006) surge da necessidade de: disponibilizar acesso aos dados para que possam ser reutilizados da maneira mais fácil possível, prover uma maneira de descobrir dados relevantes dentro de um conjunto extremamente grande de informações, e permitir a integração dos dados entre várias fontes diferentes, inclusive as que ainda não existem.

As páginas HTML são orientadas a texto e não a dados especificamente, tornando difícil o processo de extrair informações. Foram propostos vários microformatos, que são convenções bem definidas para determinados tipos de entidades, como pessoas. Porém, esses formatos são inadequados para compartilhar dados arbitrários na Web, já que é praticamente impossível registrar relações entre entidades dessa forma. Existem

também as Web APIs[1], que proveem os dados com interligações em formatos bastante conhecidos, como o XML e o JSON. Porém, as Web APIs dificultam a descoberta dos itens pelos motores de busca por não terem um equivalente à âncora do HTML e por terem escopo local.

As subseções a seguir definem uma maneira eficiente de resolver o problema da interligação entre os dados distribuídos presentes na Web de forma padronizada e formas de consultar essas informações de maneira eficiente.

2.2.1 Ontologias

Existem diferentes definições sobre o que é uma ontologia, desde formalizações entre nomenclaturas e hierarquias sobre objetos, até descrições lógicas sobre domínios. (GRUBER, 1993) a define como "uma especificação explícita de uma conceitualização consensual de um domínio". Isso significa que a ontologia age como uma padronização de tudo o que está ligado a um domínio, como seu vocabulário, fatos e regras associadas.

Uma ontologia de dados[2] é um conjunto de conhecimentos sobre dados, incluindo os fatos representados e suas relações dentro de um determinado domínio. As ontologias de domínio modelam um domínio em particular. Para isso, todos os termos associados ao domínio têm sua semântica representada explicitamente no domínio, para que no final todos os objetos e relações constituam um modelo formal do que é o próprio domínio.

Como exemplo, para definir o que é uma ontologia de música, primeiro deve-se verificar quais são todos os aspectos relacionados à musica (artistas, álbuns, compositores, instrumentos, arranjos etc, ou seja, os sujeitos e objetos), em seguida todas as relações possíveis entre eles. Isso é o vocabulário de predicados da ontologia.

Ontologias são essenciais em aplicações que desejam agregar informações de diferentes comunidades. Apesar das DTDs XML e XML Schemas serem suficientes para trocar informações entre grupos que concordaram sobre definições, sua carência em semântica impede o correto funcionamento dessa tarefa para novos vocabulários XML. Sendo assim, as ontologias têm uma maior escalabilidade, suportando novas regras, enquanto essas outras estruturas teriam que ser refeitas para suportá-las.

A importância do conceito de ontologia para este trabalho é que os dados que são consultados estão representados seguindo as regras da ontologia completa da DBPedia, aproveitando os predicados que têm significado temporal. Portanto, é essencial conhecer as regras que definem a representação dos dados que se deseja consultar.

2.2.2 O modelo RDF

O RDF[3] (*Resource Description Framework*), conforme definido em (Klyne et al. 2004) é um modelo de dados simples baseado em grafo dirigido rotulado nos nodos e arcos, projetado para uso no contexto da Web. Um recurso, ou *dataset*, aqui chamado de

1 http://en.wikipedia.org/wiki/Web_API

2 http://www.w3.org/TR/owl-features/

3 http://www.w3.org/RDF/

conjunto de dados, é representado como uma série de triplas, que são compostas de um sujeito, um predicado e um objeto, simulando a representação de uma frase em linguagem natural.

Esse modelo permite descrever entidades como pessoas, locais e objetos, de uma maneira simples e flexível, e com um potencial de ligação superior ao de documentos HTML: ao invés de interligar apenas documentos, é possível interligar as próprias entidades que, inclusive, podem pertencer a diferentes fontes de dados. A superioridade se deve ao fato das ligações não terem semântica implícita. A ligação tem um nome e dá significado literal à relação entre os dados.

Os princípios dos dados interligados, que são seguidos por esse modelo, são definidos em (BERNERS-LEE, 2006) com as seguintes regras:

- Usar URIs (*Uniform Resource Identifiers*) - para os nomes das entidades, não só para documentos, mas também para objetos do mundo real e conceitos abstratos;
- Usar URIs com o protocolo HTTP, para que seja possível visualizar os nomes;
- Quando alguém visualiza uma URI, prover informação útil, utilizando os padrões (RDF,SPARQL);
- Incluir ligações com outras URIs, para que se possa descobrir mais coisas.

O sujeito é a URI que define o recurso descrito. O objeto pode ser um literal (uma string, uma data, um número...) ou a URI de outro recurso que está relacionado ao sujeito. O predicado também fica no formato de URI e ele vem de um vocabulário definido na ontologia. Isto significa que um conjunto de URIs pode ser utilizado para representar informações sobre um determinado domínio.

O sujeito e o objeto são mapeados para nodos no grafo, enquanto que os predicados são arcos orientados entre nodos. Como exemplo, observe as informações a seguir:

“John knows Paul.”

“Marie was born in 1990.”

“John is Marie’s friend.”

“Sophia’s email is sophia@hostname.com”

Supõe-se que o servidor onde estão os dados é http://www.exemplo.org/, e o vocabulário está no mesmo local e contém os predicados {*knows*, *birthYear*, *friendOf* e *email*}. A Figura 2.1 mostra como é representado o conjunto de dados em RDF que representa corretamente essas informações:

<http://www.exemplo.org/John> <http://www.exemplo.org/knows> <http://www.exemplo.org/Paul> .

<http://www.exemplo.org/Marie> <http://www.exemplo.org/birthYear> “1990” .

<http://www.exemplo.org/John> <http://www.exemplo.org/friendOf> <http://www.exemplo.org/Marie> .

<http://www.exemplo.org/Sophia> <http://www.exemplo.org/email>"sophia@hostname.com”

Figura 2.1. Exemplo de conjunto de triplas RDF.

O grafo que representa esse conjunto de dados é ilustrado na Figura 2.2 (assume-se url como http://www.exemplo.org):

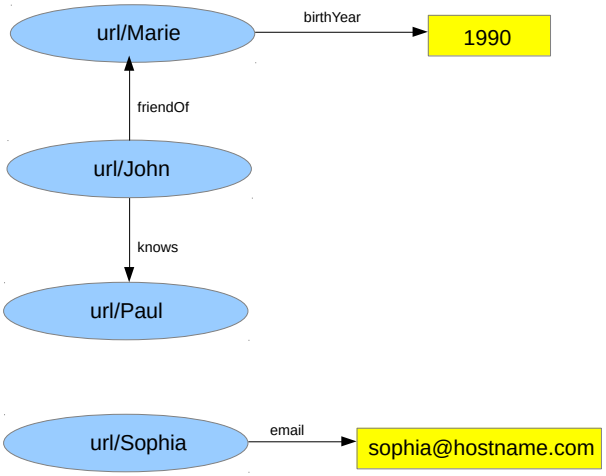

Figura 2.2. Exemplo de grafo associado às triplas RDF.

Tendo em vista que o RDF não é um formato de dados, mas apenas um modelo para descrever recursos no formato "sujeito predicado objeto", para publicar um grafo de RDF na Web é necessário serializá-lo. Existem diversos formatos de serialização. Um deles é o Turtle (BECKETT et. al, 2008). Mais especificamente, neste trabalho é utilizada a variante, N3, que difere do Turtle apenas por não utilizar os *namespaces* para prefixos. O motivo disso é que *namespaces* não são realmente necessários já que os termos da DBPedia são representados por URIs que, quando acessadas por HTTP, fornecem detalhes sobre o que são. Explicitar isso no conjunto de dados RDF causaria muita repetição desnecessária, o que seria especialmente negativo em um conjunto de dados tão extenso como esse.

As figuras 2.3 e 2.4 ilustram exemplos de conjuntos de triplas RDF nos formatos Turtle e N3, respectivamente:

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

@prefix foaf: <http://xmlns.com/foaf/0.1/> .

<http://biglynx.co.uk/people/dave-smith>

rdf:type foaf:Person ;

foaf:name "Dave Smith" .

Figura 2.3. Exemplo de conjunto de triplas RDF em Turtle.

Exemplo de N3:

<http://biglynx.co.uk/people/dave-smith> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .

<http://biglynx.co.uk/people/dave-smith> <http://xmlns.com/foaf/0.1/name> "Dave Smith" .

Figura 2.4. Exemplo de conjunto de triplas RDF em N3.

Como pode ser visto na figura acima, o formato de serialização utilizado nesse trabalho separa as triplas através de um ponto final (.), e os demais elementos aparecem entre delimitadores iguais aos de *tags*, (< e >).

2.2.3 SPARQL

SPARQL, ou *Simple Protocol and RDF Query Language*, é uma linguagem de consulta para o RDF. Sua sintaxe é bastante intuitiva e simples, muito semelhante à

sintaxe do SQL, mas bastante poderosa devido à riqueza das informações que é capaz de recuperar a partir das triplas normalizadas. SPARQL possibilita que o programador envolva diferentes fontes de dados e de vocabulários da ontologia, independente de onde ou como estejam hospedadas, referenciando-as facilmente através de sua URI.

Um *endpoint* SPARQL - em protocolos de comunicação, *endpoint* é o nome dado para uma entidade em um terminal de uma conexão - aceita consultas e retorna os resultados via HTTP, em forma de tabelas onde cada linha corresponde a uma tripla encontrada nos conjuntos de dados. A DBPedia disponibiliza o seu *endpoint* em http://dbpedia.org/sparql, onde é possível escolher o formato da saída (HTML, XML, CSV e outros).

Em uma consulta padrão, definem-se primeiramente os prefixos, que nada mais são que uma comodidade para não repetir os mesmos trechos de URI várias vezes. Na consulta em si, têm-se, do mesmo modo que em uma consulta SQL comum, a forma SELECT-FROM-WHERE.

Primeiramente, a cláusula FROM determina qual, ou quais os conjuntos de dados envolvidos na consulta. De forma mais simples, é a maneira de indicar ao motor de busca do SPARQL onde estão localizadas as triplas RDF que devem ser pesquisadas. Para agregar diferentes arquivos de triplas, pode-se uní-las em um conjunto de dados que engloba todas as triplas de interesse ou pode-se enumerar todas as fontes de dados na própria cláusula (a exemplo do SQL). Neste último caso, deve ser utilizado FROM NAMED para que cada diferente fonte seja identificada. O grafo RDF, a partir do qual são feitas as consultas, é obtido a partir da agregação dos grafos individuais. Caso haja um conjunto de dados padrão, a cláusula FROM pode ser ocultada, sendo que tal conjunto representa o grafo para consulta.

A cláusula WHERE indica simplesmente qual a regra que define se cada uma das triplas t (sujeito, predicado, objeto) pertencentes ao conjunto de dados indicado deve ou não aparecer no resultado. Este é um procedimento semelhante ao do SQL, mas além de utilizar comparações com literais e entre elementos, é possível incluir relações em uma sintaxe de *pattern matching*[4], o que certamente enriquece consideravelmente a particularidade do que está sendo selecionado nos dados.

Finalmente, a cláusula SELECT é a mais simples de todas: ela somente indica, dentre tudo o que foi selecionado no passo anterior, quais os campos que devem ser projetados para o resultado final, normalizando-os e formatando-os conforme a necessidade de exibição. As variáveis inicializam com um ponto de interrogação (?) e definem o que seriam os caracteres curinga do *pattern matching*. Também é possível definir filtros, ordenações, campos opcionais e limite sobre a seleção das triplas.

Supondo-se que haja um conjunto de dados de livros representados em RDF no arquivo livros.nt e um conjunto de relatórios no arquivo relatorios.nt:

4 Técnica *utilizada para* verificar se um texto se encaixa na estrutura de um padrão definido formalmente.

PREFIX livros: <http://www.servidor1.com.br/livros>

PREFIX relatórios: <http://www.relatorios.net/publicados>

PREFIX pred: <http://www.servidordepredicados.com/predicados_de_textos>

PREFIX autores: <http//www.autoresdomundo.com/autores>

SELECT ?titulo ?autor

FROM NAMED <livros.nt>

FROM NAMED <relatorios.nt>

WHERE ?titulo pred:writtenBy ?autor

Figura 2.5. Exemplo de consulta SPARQL sobre um conjunto de dados RDF

A consulta ilustrada na Figura 2.5 inicia criando prefixos ou *aliases* para as URIs que descrevem a localização das entidades. É importante notar que não há problema algum em interligar entidades localizadas em diferentes servidores: isso é transparente para a consulta e garante uma grande escalabilidade.

É definido que as triplas dos dois arquivos são lidas sequencialmente para a construção do grafo final. Em seguida, são selecionadas somente as triplas onde o predicado utilizado é writtenBy. Por fim, são retornados os pares (título, autor) dessas triplas. Isto é, o resultado é uma tabela onde a coluna da esquerda é o título do livro ou do relatório e a coluna da direita é o autor.

Considere o conjunto de triplas[5] RDF referente ao arquivo sobre livros e relatórios, respectivamente nas figuras 2.6 e 2.7.

livros.nt:

<livros/Dom_Casmurro> <pred/writtenBy> <autores/Machado_de_Assis> .
<livros/Dom_Casmurro> <pred/publicationYear> "1899" .
<livros/Quincas_Borba> <pred/writtenBy> <autores/Machado_de_Assis> .
<livros/Os_Escravos> <pred/writtenBy> <autores/Castro_Alves> .
<livros/Os_Escravos> <pred/publicationYear> "1883" .
<livros/O_Guarani> <pred/writtenBy> <autores/Jose_de_Alencar> .

Figura 2.6. Conjunto de triplas RDF do conjunto de dados dos livros.

relatorios.nt:

<relatorios/Exemplo_4> <pred/writtenBy> <autores/John_Nameless> .
<relatorios/Estagio_Xyz> <pred/writtenBy> <autores/Marie_Watson> .
<relatorios/Estagio_Xyz> <pred/numberOfPages> "51" .
<relatorios/Experimento_8> <pred/writtenBy> <autores/John_Nameless> .
<relatorios/Experimento_8> <pred/writtenBy> <autores/Sophia_Test> .
<relatorios/Experimento_8> <pred/publishedAt> "Porto Alegre" .

Figura 2.7. Conjunto de triplas RDF do conjunto de dados dos relatórios.

O retorno da consulta SPARQL ilustrada na Figura 2.5 executada sobre esses dados teria o mesmo conteúdo que o representado pela tabela abaixo:

5 Por questão de legibilidade, foram utilizados os mesmos prefixos da Figura 2.5, apesar do N3 não permitir o uso de prefixos.

Tabela 2.1. Exemplo de retorno de consulta SPARQL

titulo	*autor*
http://www.servidor1.com.br/livros/Dom_Casmurro	*http//www.autoresdomundo.com/autores/Machado_de_Assis*
http://www.servidor1.com.br/livros/Quincas_Borba	*http//www.autoresdomundo.com/autores/Machado_de_Assis*
http://www.servidor1.com.br/livros/Os_Escravos	*http//www.autoresdomundo.com/autores/Castro_Alves*
http://www.servidor1.com.br/livros/O_Guarani	*http//www.autoresdomundo.com/autores/Jose_de_Alencar*
http://www.relatorios.net/publicados/Exemplo_4	*http//www.autoresdomundo.com/autores/John_Nameless*
http://www.relatorios.net/publicados/ Estagio_XYZ	*http//www.autoresdomundo.com/autores/Marie_Watson*
http://www.relatorios.net/publicados/Experimento_8	*http//www.autoresdomundo.com/autores/John_Nameless*
http://www.relatorios.net/publicados/Experimento_8	*http//www.autoresdomundo.com/autores/Sophia_Test*

2.3 Ferramentas

A ferramenta utilizada para este trabalho foi o Joseki, que é uma parte do Jena. A seguir, essas duas ferramentas são explicadas com mais detalhes.

Jena é um framework desenvolvido em Java pela *HP Labs Semantic Web Research*[6] e se destina a construir aplicações para a Web Semântica. Jena provê um ambiente para programação com RDF, OWL[7] e SPARQL. É um projeto em código aberto que inclui uma API para o RDF, leitura e escrita de RDF nos formatos RDF/XML, N3 e N-triples, uma API para o OWL, armazenamento persistente em memória, e um servidor que constrói grafos e realiza consultas SPARQL dobre dados, chamado Joseki.

O Joseki provê uma funcionalidade para consultas SPARQL sobre conjuntos de dados locais representados como conjuntos de dados RDF, excelente para projetos de pequeno porte. Joseki disponibiliza um serviço HTTP que funciona como motor de busca sobre o grafo RDF armazenado na memória, obtido a partir do processamento dos conjuntos de dados pelo servidor. Joseki roda em qualquer dispositivo com a máquina virtual Java, e disponibiliza alguns serviços, como, por exemplo, a consulta SPARQL sobre o grafo e a atualização online do grafo, que não é utilizada para fins deste trabalho. Joseki deixa ainda o usuário criar novos serviços, através da sintaxe do RDF ou OWL.

O servidor RDF Joseki possibilita:

- definir conjuntos de dados RDF pré-estabelecidos a partir de arquivos locais ou remotos;
- carregar os conjuntos de dados para a memória em forma de um grafo RDF;
- consultar, em linguagem SPARQL, o grafo do conjunto de dados atualmente carregado; e
- atualizar o grafo com novas triplas RDF.

6 http://www.hpl.hp.com/semweb/

7 Web Ontology Language, uma linguagem para definir as ontologias

Os três primeiros conceitos (definir, carregar e consultar) são explicados em detalhes a seguir porque foram utilizados na implementação da ferramenta proposta neste trabalho.

Os serviços são descritos no arquivo de configuração joseki-config.ttl, o qual é escrito em RDF. Neste arquivo, são definidos todos os conjuntos de dados que o servidor conhece. Um conjunto de dados é definido através de um rótulo para o conjunto, um rótulo para cada arquivo RDF que compõe o comjunto, e a localização de cada um desses arquivos (caminho do arquivo local ou URI do arquivo remoto). Um exemplo de conjunto de dados definido para livros é ilustrado na Figura 2.8

```
<#books>   rdf:type ja:RDFDataset ;
  rdfs:label "Books" ;
  ja:defaultGraph
    [ rdfs:label "dados.nt" ;
      a ja:MemoryModel ;
      ja:content [ja:externalContent <file:Data/dados.nt> ] ;
    ] ;
```

Figura 2.8. Exemplo de definição de um conjunto de dados no Joseki

Sempre que o servidor é iniciado, cada um dos conjuntos de dados presentes no arquivo de configuração é processado, para obter o respectivo grafo RDF que representa suas triplas, e, em seguida, o grafo é carregado na memória.

Os serviços de interesse são os serviços que possibilitam consultas SPARQL sobre os grafos. Deve-se dar um nome ao serviço e especificar sobre qual grafo as consultas são feitas. A Figura 2.9 mostra um exemplo de serviço sobre o conjunto de dados dos livros criado e ilustrado na Figura 2.8:

```
<#service2>
  rdf:type          joseki:Service ;
  rdfs:label        "SPARQL on big ammounts of data" ;
  joseki:serviceRef   "books" ;
  joseki:dataset      <#books> ;
  joseki:processor    joseki:ProcessorSPARQL_FixedDS ;
```

Figura 2.9. Exemplo de definição de um serviço no Joseki

Nesse tipo de serviço, o conjunto de dados é fixo (no exemplo, *books*), portanto as consultas SPARQL não podem conter a cláusula WHERE. Uma alternativa seria não especificar o conjunto de dados e deixar para que o usuário o forneça na própria consulta. Entretando, esse procedimento pode ser menos eficiente porque o grafo precisa ser gerado somente após a submissão da consulta. A vantagem é o aumento da flexibilidade já que um serviço não estaria preso a um conjunto de dados específico.

Com o servidor Joseki online, os serviços podem ser utilizados facilmente através de um navegador, utilizando-se o protocolo HTTP e passando a consulta SPARQL como um argumento GET da própria URL acessada. Por exemplo, para consultar no

grafo de livros da Figura 2.8 a partir de um servidor Joseki rodando localmente na porta 2020, bastaria acessar o seguinte endereço:

http://localhost:2020/books?query=CONSULTA&output=xml&stylesheet=%2Fxml-to-html.xsl

O resultado é retornado na forma de um arquivo XML, mas para a exibição direta na tela é solicitado ao servidor que converta o estilo de exibição de XML para HTML. Existem outros formatos nos quais os resultados poderiam ser dados, como: HTML, planilha, JSON, Javascript, N-triples, RDF/XML ou CSV.

2.4 Considerações Finais

Neste capítulo, foram apresentados conceitos e ferramentas utilizados durante o trabalho. A ferramenta proposta responde perguntas relacionadas ao tempo. Para ela funcionar, é necessário que as informações com conteúdo temporal presentes nos arquivos da Web sejam capturadas e armazenadas em um formato padronizado de dados interligados compatível com o da DBPedia.

Para isso, é necessário entender a ontologia que define os fatos e regras associadas ao domínio da DBPedia, pois as URIs e predicados devem ser conhecidos para que se possa fazer a consulta sobre o conjunto de dados, armazenado através do modelo RDF. As consultas são feitas utilizando o protocolo SPARQL que verifica a existência de triplas no grafo obtido a partir do conjunto de dados que satisfazem um determinado padrão.

A ferramenta utilizou a ferramenta Joseki, que é provida pelo framework Jena. O servidor do Joseki provê um serviço de consultas SPARQL que pode ser chamado localmente através de um navegador, e os resultados podem ser exibidos no próprio navegador ou salvos como um arquivo XML, por exemplo.

3 TRABALHOS RELACIONADOS

Esse capítulo apresenta os principais trabalhos relacionados à ferramenta desenvolvida neste trabalho. Os trabalhos relacionados estão divididos em duas categorias. A primeira categoria descreve os trabalhos que têm relacionamento direto com o objetivo da ferramenta proposta, que é extrair e consultar informações temporais. A segunda categoria descreve os trabalhos que interpretam e respondem consultas em linguagem natural.

A empresa alemã Neofonie[8], que teve início dentro da Universidade Técnica de Berlin, desenvolveu uma aplicação de busca facetada sobre dados da Wikipedia. Utilizando-se de dados estruturados em RDF das ontologias da DBPedia, foi implementado um sistema de busca que suporta consultas mais complexas do que a maioria dos motores de busca convencionais: seu funcionamento se baseia em uma pesquisa incremental. Primeiro, o usuário escolhe um tipo de dado inicial. Em seguida, são carregados todos os predicados associados ao predicado em forma de triplas RDF, na forma de filtros para restringir mais ainda a solução. Certamente, esta é uma maneira interessante de pesquisar dados, não só porque não depende tanto da associação de palavras-chave, mas também porque possibilita uma descoberta de informações que o usuário não havia previsto, mas que estava relacionada ao domínio de sua pesquisa. A ferramenta proposta neste trabalho, apesar de também utilizar os dados da DBPedia, difere por tratar uma consulta em linguagem natural e ser específica para consultas com relevância temporal.

O trabalho proposto por (GALLINA, 2011a) consiste em identificar, extrair e normalizar informações temporais em páginas da Web na língua inglesa, e a partir disso organizar semanticamente essa informação, em forma de dados estruturados. Esse procedimento de extração e organização semântica é denominado EXTIO (*Extraction of Temporal Information Using Ontologies*). O processo de identificação e extração dos termos temporais é feito através da definição de uma gramática formal para um grande conjunto de expressões em inglês (GALLINA, 2011b). Sendo assim, é feito um processamento sobre essas páginas para converter cada uma dessas ocorrências em um formato de data padronizado. Tendo o texto com as expressões temporais relativas normalizadas, os fatos temporais são extraídos e organizados em RDF, no formato N3, utilizando a ontologia da DBPedia e extendendo seu conjunto de informações. Esse trabalho, EXTIO, foi a motivação para a ferramenta desenvolvida neste trabalho. A ferramenta aqui proposta é complementar ao EXTIO, pois tendo a informação temporal representada em um conjunto de dados compatível com o da DBPedia, faz-se necessária

8 http://www.neofonie.de/

uma ferramenta para realizar consultas temporais sobre esses dados, que é o que foi realizado neste trabalho.

O trabalho de (MANICA, 2010) apresenta uma abordagem, chamada de *Two Phase Interception*, que possibilita o suporte a consultas temporais por palavras-chave em documentos XML. O trabalho também propõe uma classificação genérica para servir de guia para mecanismos de busca temporal por palavras-chave. Essa classificação convenciona um formato para as consultas para que possam ser adequadamente mapeadas para a entidade temporal, podendo ser um valor absoluto ou um intervalo de tempo. O formato básico utilizado é: o sujeito seguido da palavra-chave temporal, que vai definir qual regra aplicar à entidade temporal fornecida, que é o último termo da consulta. Segundo o autor, não há prejuízo à intuitividade para o usuário em formatar a consulta dessa forma, pois estudos apontam que a maioria das consultas temporais segue um formato semelhante a esse. A principal diferença com relação a este trabalho é que a ferramenta aqui apresentada trabalha com as informações representadas no formato de dados interligados. Porém, o formato utilizado para pesquisas e retornos aproveitou algumas das convenções adotadas nesse trabalho relacionado.

A DBPedia[9] é um projeto comunitário, que tem por objetivo extrair as informações da Wikipedia[10], transformá-las em um formato de dados estruturados, e disponibilizar essa informação na Web para consultas mais sofisticadas. Também é possível interligar dados de outras fontes com os dados da Wikipedia e obter novos resultados graças à informação implícita contida nas relações entre essas fontes.

O projeto iniciou em 2007 e tem em sua base de conhecimento informações sobre mais de 3,64 milhões de coisas, das quais 1,83 milhões são classificadas a partir de uma grande ontologia consistente (a ontologia da DBPedia). Seu conjunto de dados contém mais de 1 bilhão de triplas RDF, das quais 385 milhões foram extraídas a partir da versão em inglês da Wikipedia. Muitas vezes, essa coleção é referenciada na literatura como a "Web de dados" ou "Web Semântica". Da mesma forma que é possível para qualquer autor contribuir incluindo ou editando informações da Wikipedia, também é possível contribuir com o projeto DBPedia, com novas triplas que utilizam os predicados definidos na ontologia da DBPedia, estendendo a "Web Semântica" e, consequentemente, auxiliando o reuso para os próximos colaboradores. É assim que o projeto consegue evoluir e proporcionar essa imensa variedade de informações em um formato simples e fácil de manipular. A ferramenta proposta neste trabalho realiza as pesquisas a partir da ontologia da DBPedia, considerando apenas as propriedades com valor temporal importantes. Sendo assim, esses dados representados na DBPedia que têm tempo associado podem ser utilizados para responder perguntas feitas pelo usuário da ferramenta.

O trabalho de (PUSTEJOVSKY, 2005) contém um estudo sobre a informação de tempo e eventos na linguagem natural, em especial no contexto de sistemas de perguntas e respostas (sistemas QA). O trabalho verifica diferentes aspectos do tema como a ordenação temporal de eventos, e apresenta uma linguagem, TimeML, com o objetivo de capturar esse tipo de elementos na linguagem. O trabalho define diferentes tipos de perguntas temporais e uma termologia para eles. A partir dos termos contidos na frase e da própria informação nela fornecida, é possível identificar a qual classe de

9 http://dbpedia.org

10 http://en.wikipedia.org/wiki/Main_Page

perguntas a frase pertence. Por exemplo, há perguntas mais simples, em que se deseja saber quando um evento bem definido aconteceu ou o que aconteceu durante uma determinada data. Há também questões mais complexas, como perguntar o que aconteceu durante um evento que se repete com uma determinada frequência (como o Ramadan), mas restringindo isso para um ano específico. A ferramenta aqui apresentada faz uso da definição de perguntas contida nesse trabalho relacionado, com algumas modificações e simplificações, o que serve como base para a implementação do módulo de identificação da pergunta da ferramenta.

Uma ferramenta de QA existente na Web é a *TrueKnowledge*[11], que possui muitas semelhanças em relação à implementação da ferramenta proposta neste trabalho. *TrueKnowledge* visa prover ao usuário uma maneira de acessar a imensa quantidade de informação de uma maneira simples e natural. Através de perguntas em linguagem natural, como se estivessem falando com outro humano, os usuários submetem questões sobre algum tema e o sistema responde a eles em linguagem natural. Ao invés de simplesmente retornar uma lista de *links* que pode não ser relevante à questão em si, o sistema identifica com precisão o que o usuário perguntou e responde apenas aquela informação útil, citando uma referência (se houver) de onde ou de que contexto a informação é proveniente, algo semelhante a um "leia mais". No momento desta publicação, *TrueKnowledge* contava com uma base de conhecimento com mais de 635 milhões de fatos e 27 milhões de coisas e, segundo os desenvolvedores, guarda essas informações de uma maneira suficientemente estruturada para que possa ser lidada eficientemente pelo sistema, mas simples o bastante para ser entendida e submetida por voluntários. O fato de a resposta ser automática através de inferências sobre os dados existentes, ao invés dos sites de QA com intervenção humana, garante um grande avanço na área. A diferença principal entre a *TrueKnowledge* e a ferramenta proposta neste trabalho é que enquanto a primeira tem como objetivo ser algo bastante amplo e genérico, isto é, não trata informação temporal de modo essencial, a ferramenta proposta neste trabalho se destina justamente a mostrar como é possível um aproveitamento alto de todas informações temporais obtidas nos documentos. Pode-se constatar que, enquanto a aplicação de QA consegue responder perguntas temporais simples, como quando aconteceu um evento bem específico, quando pergunta-se algo mais complexo que exige um processamento dos dados temporais, ou quando o que se deseja saber não é o tempo em si, mas alguma informação relativa ao tempo, a qualidade da resposta é ruim.

11 http://www.trueknowledge.com/

4 FERRAMENTA DE BUSCA TEMPORAL

Esse capítulo descreve a ferramenta de busca temporal desenvolvida neste trabalho, dando uma visão sobre seu funcionamento. A organização do capítulo se dá da seguinte maneira: na seção 4.1, é dada uma visão geral sobre o funcionamento da ferramenta. Na seção 4.2 são detalhadas as informações técnicas, requisitos e instruções para a utilização da ferramenta. A seção 4.3 define formalmente a sintaxe e a semântica dos tipos de perguntas suportados pela ferramenta. A seção 4.4 define quais são as propriedades da DBPedia suportadas pela implementação. A seção 4.5 detalha o funcionamento de todos os módulos da ferramenta que foram apresentados na seção 4.1. A seção 4.6 apresenta o funcionamento da ferramenta através de capturas de tela com alguns casos de teste. A seção 4.7 é um resumo geral do capítulo.

4.1 Visão Geral

A ferramenta desenvolvida neste trabalho consiste em responder perguntas, em linguagem natural na língua inglesa, com relevância temporal e associadas a um conjunto de dados específico representado através da estrutura de dados interligados RDF, e com as regras que definem seus sujeitos e predicados definidos em uma ontologia, especificamente a da DBPedia.

A primeira parte do funcionamento normal da aplicação consiste em apresentar a tela inicial onde é aguardada uma pergunta. Quando a pergunta é submetida, faz-se a leitura. O próximo passo é identificar devidamente em qual situação a pergunta se encontra, ou seja, qual o padrão de pergunta utilizado, conforme será explicado nas próximas seções. Adicionalmente, identifica-se o que já é sabido e o que deve ser inferido para responder à pergunta. Isso é feito através da análise sintática da pergunta, que também será explicada adiante.

Tendo caracterizado o tipo da pergunta, as informações devem ser validadas. Quando isso acontece, inicia-se o processo de normalização e tradução da pergunta do usuário para uma consulta SPARQL. A consulta deve, ao mesmo tempo: (i) abranger mais do que o usuário perguntou, possibilitando a descoberta de informações que respondem à pergunta, mas que podem estar representadas de uma forma não equivalente à consulta, e (ii) priorizar o conteúdo exato que foi solicitado, como é o esperado pelo usuário.

A seguir, a consulta é executada sobre o conjunto de dados local, utilizando o serviço de consultas SPARQL provido pelo Joseki, e os resultados referentes às triplas pertencentes ao resultado da consulta, se houverem, são convertidos para um texto, também em linguagem natural, que responde à pergunta feita pelo usuário.

O processo visa ser ao mesmo tempo simples, direto e amigável para o usuário, e também poderoso. As respostas não sofrem nenhuma espécie de ranqueamento no que diz respeito à frequência em que aparecem, isto é, cada tripla do conjunto de dados tem a mesma importância para o processo de resposta. Entretanto, em algumas situações são aplicadas algumas regras quanto à ordem em que aparecem nos resultados, visando priorizar os resultados mais similares à pergunta original ou apresenta-los de forma organizada.

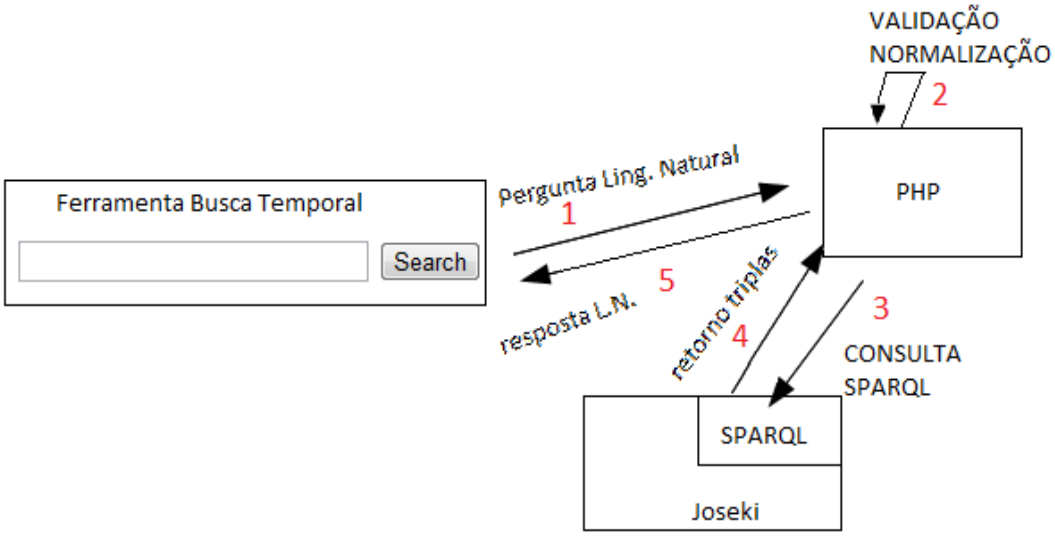

Figura 4.1. Funcionamento geral da ferramenta.

A Figura 4.1 é um esboço do fluxo de execução que foi explicado. Todo o processo é transparente para o indivíduo que utiliza o sistema. Nas seções seguintes, serão mostradas as capturas de tela que evidenciam a usabilidade da aplicação em diferentes casos.

4.2 Informações e requisitos técnicos

Essa seção se destina a explicar quais linguagens e ferramentas foram utilizadas para o desenvolvimento, quais particularidades ou limitações técnicas existem, e quais decisões técnicas foram tomadas para a implementação da ferramenta.

A implementação da ferramenta de busca temporal foi feita utilizando a linguagem PHP[12]. A aplicação foi desenvolvida rodando localmente através do WampServer[13] versão 2.2, rodando a versão 5.3.8 do PHP. Ao todo foram desenvolvidos oito arquivos PHP e um total de 1287 linhas de código.

Através de um formulário HTML presente no script PHP principal, é submetida a pergunta. A normalização, determinação do tipo de pergunta e conversão para consulta SPARQL também é feita na mesma linguagem.

Depois disso, é chamado um serviço SPARQL, que roda no servidor Joseki, versão 3.4.4. Para rodar corretamente, é necessário ter o Java Runtime Environment (JRE).

12 Hypertext Preprocessor

13 WampServer é um software para Windows que contém o PHP, MySQL e o Apache.

Recomenda-se a versão mais recente, que foi a utilizada nesse trabalho. O servidor roda em qualquer sistema operacional que rode o Java, como o Linux e o Windows. A Figura 4.2 ilustra quais os comandos que devem ser executados no prompt de comando para rodar o servidor do Joseki no Microsoft Windows:

```
cd {diretório do Joseki}
set JOSEKIROOT={diretório do Joseki}
bin\joseki_path
set CLASSPATH=%CP%
echo %CLASSPATH%
bin\rdfserver
```

Figura 4.2 Comandos de prompt para rodar o Joseki

O arquivo de dados com as triplas em formato RDF deve estar no formato N3. Para garantir que o servidor esteja adequadamente configurado para servir os dados RDF no formato N3, pode ser necessário introduzir a seguinte diretiva no arquivo httpd.conf do Apache (ou diretamente no diretório desejado, no arquivo .htaccess):

```
AddType text/n3; charset=utf-8 .nt
```

Todas as bibliotecas e funções utilizadas para desenvolver a ferramenta já se encontravam nativas no próprio PHP. Para funcionar corretamente, além das configurações padrão da instalação, a opção *short open tag* deve estar ativa no servidor Apache.

Como a DBPedia e sua ontologia, bem como a grande maioria dos dados nela armazenados encontram-se em inglês, essa ferramenta foi completamente desenvolvida com suporte apenas a esse idioma, de modo que a interface, os textos de ajuda, a entrada e a saída, e as convenções sintáticas e semânticas estão todas de acordo com esse idioma.

O serviço HTTP do Joseki, denominado *temporal*, executa as consultas SPARQL para essa ferramenta, e pode ser chamado pelo navegador a partir do seguinte endereço:

```
http://localhost:2020/temporal?query=consulta&output=saida&stylesheet=estilo
```

A saída escolhida para a ferramenta foi o XML e o estilo foi “XML to HTML”. A Figura 4.3 ilustra a maneira como foi criado o serviço *temporal* no Joseki:

```
<#temporal>
   rdf:type          joseki:Service ;
   rdfs:label        "Temporal Search Tool" ;
   joseki:serviceRef   "temporal" ;
   # dataset part
   joseki:dataset      <#temporaldata> ;
   joseki:processor    joseki:ProcessorSPARQL_FixedDS ;
   .
```

Figura 4.3 Definição do serviço temporal

O conjunto de dados, temporaldata, referenciado na Figura 4.3, foi definido como ilustrado na Figura 4.4:

```
<#temporaldata>   rdf:type ja:RDFDataset ;
  rdfs:label "Temporal Data" ;
  ja:defaultGraph
    [ rdfs:label "dados.nt" ;
      a ja:MemoryModel ;
      ja:content [ja:externalContent <file:Data/dados.nt> ] ;
    ] ;
  .
```

Figura 4.4 Definição do conjunto de dados temporaldata

Pode-se notar na Figura 4.3 que o conjunto de dados é fixo no serviço *temporal*. Isso implica no fato da cláusula WHERE das consultas SPARQL ser implícita, não fazendo parte das consultas.

O arquivo com o conjunto de dados utilizado para os testes foi um subconjunto do conjunto de dados *Raw Infobox Properties*, da DBPedia. O recorte feito no arquivo foi totalmente arbitrário e teve como finalidade apenas reduzir o tamanho total do arquivo, de 1GB para 134MB. Mesmo assim, restaram 1.010.882 triplas. A maior parte das informações desse conjunto é sobre pessoas famosas e eventos ligados ao exército e à política dos Estados Unidos, obtidos principalmente a partir da Wikipedia. A ferramenta desenvolvida funciona com qualquer arquivo de dados em N3 que siga as regras da ontologia da DBPedia, sendo que basta criar um novo serviço no Joseki para prover um *endpoint* de consultas SPARQL para o arquivo desejado, ou até mesmo jogá-lo no lugar do arquivo de dados do serviço *temporal*, sobrescrevendo o arquivo dados.nt.

4.3 Perguntas temporais

Esta seção se subdivide em duas subseções: na subseção 4.3.1, é dada a definição e semântica de cada tipo de pergunta temporal suportada pela ferramenta, juntamente com as suas siglas. Na subseção 4.3.2, é descrita a sintaxe requerida para as perguntas, ou seja, os formatos que as perguntas devem seguir em cada caso.

4.3.1 Perguntas Suportadas

Baseando-se fortemente nas definições de tipos de perguntas temporais de (PUSTEJOVSKY, 2005), foram estabelecidos nove tipos de perguntas com relevância temporal. Trata-se de perguntas que uma pessoa faria no cotidiano em linguagem natural sobre algum assunto. Acredita-se que essas categorias de questões, levando em conta a estrutura da DBPedia para representação de fatos temporais, sejam suficientes para responder o máximo de perguntas feitas sobre os dados existentes.

A Tabela 4.1 apresenta essas perguntas, suas siglas, e o que elas representam. A maneira como cada uma delas acessa os dados para efetivamente responder às questões recebidas é apresentada em detalhes na subseção 4.5.3. Na Tabela 4.1, as palavras entre chaves significam informações opcionais.

Tabela 4.1 Perguntas temporais: semântica

Sigla	**Semântica da pergunta**	**Dados fornecidos**	**Dados desejados**
Q_T-E	Quando o sujeito praticou o evento	Sujeito, evento	Tempo
Q_E-T	O que aconteceu no tempo	Tempo	Sujeito, evento, {tempo}
Q_I-ET	Quem praticou o evento no tempo	Evento, tempo	Sujeito (pessoa)
Q_V-ET	O que sofreu o evento no tempo	Evento, tempo	Sujeito (objeto)
Q_{YN}	Saber se o sujeito praticou ou sofreu o evento no tempo ou não	Sujeito, evento, tempo	Sim ou não
Q_E-(Q_T-E)	O que (mais) aconteceu quando o sujeito praticou o evento	Sujeito, evento	Sujeito, evento, {tempo}
Q_E-T^B	O que aconteceu entre t1 e t2	Tempo 1, tempo 2	Sujeito, evento, {tempo}
Q_I-ET^B	Quem praticou o evento entre t1 e t2	Evento, tempo 1, tempo 2	Sujeito (pessoa), {tempo}
Q_V-ET^B	O que sofreu o evento entre t1 e t2	Evento, tempo 1, tempo 2	Sujeito (objeto), {tempo}

Como pode ser visto na Tabela 4.1, fornecem-se algumas características que devem aparecer no resultado, e obtêm-se o(s) resultado(s) onde existe um casamento dessas características. De modo geral, o resultado é tudo que pertence à tripla armazenada e não foi especificado (o complemento da tripla). Entretanto, em alguns casos também são retornados alguns dados redundantes. Isso se deve ao fato de que, apesar do tempo ser fornecido na própria pergunta, ele pode ser fornecido de maneira incompleta, como em perguntas contendo apenas o ano, ou perguntas com um período entre duas datas. Nesses casos, é interessante que o resultado seja o mais completo possível, então é importante que no resultado seja informada a data completa, quando disponível, de cada tripla que pertença ao conjunto solução da pesquisa.

4.3.2 Sintaxe das Perguntas

Essa subseção consiste em definir formalmente qual a sintaxe aceita para cada uma das perguntas. Pode também ser vista como um complemento ao texto de ajuda presente dentro da própria ferramenta e uma importante documentação. As perguntas seguem o padrão formal de sintaxe da língua inglesa, com algumas simplificações. O formato de datas utilizado é o mês/dia/ano, com zeros à esquerda, e anos de quatro dígitos. Existem restrições quanto a alguns termos e no número de sinônimos suportados, visto que a quantidade de variações é extensa e o seu suporte total escaparia do escopo do trabalho.

A pergunta Q_E-(Q_T-E) funciona como uma espécie de função composta: a Q_T-E interna retorna um tempo, que é utilizado como argumento da Q_E-T externa. Assim, é possível descobrir eventos que aconteceram ao mesmo tempo em que outro determinado evento. A temporalidade nesse caso é implícita e transparente ao usuário, que não precisa saber quando aconteceu o seu fato de interesse, bastando perguntar sobre o próprio fato.

A Tabela 4.2 define a sintaxe esperada para cada tipo de pergunta, bem como alguns exemplos em cada caso. Foram feitas as seguintes convenções na tabela: x/y significa a alternativa, x ou y; um elemento entre colchetes significa que algum texto que o represente deve aparecer ali, sem os colchetes; e um elemento entre chaves é opcional.

Tabela 4.2 Perguntas temporais: sintaxe e exemplos

Sigla	**Sintaxe**	**Exemplos**
Q_T-E	*When/In what year did/was [sujeito] [evento]?*	*When was Bill Gates born?* *In what year did Aristotle die?*
Q_E-T	*What happened in/on [tempo]?*	*What happened in 1920?* *What happened on 04/21/1990?*
Q_I-ET	*Who/Whose {was} [evento] [tempo]?*	*Who died in 1944?*
Q_V-ET	*What/which {was} [evento] [tempo]?*	*What was published on 01/29/1929?*
Q_{YN}	*Was/did [sujeito] [evento] [tempo]?*	*Was Cobble Hill Tunnel added on 09/07/1989?*
Q_E-(Q_T-E)	*What happened when [sujeito] [evento]?*	*What happened when Amy Grant was born?*
Q_E-T^B	*What happened between [tempo 1] and [tempo 2]?*	*What happened between 02/17/1950 and 05/17/1950?*
Q_I-ET^B	*Who/whose {was} [evento] between [tempo 1] and [tempo 2]?*	*Who was born between 1950 and 1960?*
Q_V-ET^B	*What/which {was} [evento] between [tempo 1] and [tempo 2]*	*What was discovered between 01/01/1970 and 12/31/1972?*

Como foi verificado que os dados da DBPedia analisados se referem ao passado, não foi implementado um suporte a pesquisas temporais sobre o presente ou futuro. No entanto, isso poderia ser feito sem maiores dificuldades devido à simplicidade da mudança dos termos da frase no idioma inglês quando uma pergunta desses tipos troca de tempo verbal.

4.4 Propriedades Suportadas

Foi feito um estudo estatístico a respeito das ocorrências de propriedades, também chamadas de predicados, nos conjuntos de dados (GALLINA, 2011a). Como seria trabalhoso demais suportar todas as propriedades e muitas delas quase não aparecem, e também devido ao fato do trabalho focar em mostrar a possibilidade de consultar dados temporais, mais do que fazer uma ferramenta completa prevendo todos os casos, acredita-se que foi suficiente pegar as propriedades que ocorreram com mais frequência.

Foram separadas, dentre todas as propriedades da DBPedia, aquelas que têm relação temporal. Então, foi feita a contagem da ocorrência de cada uma delas entre 1981 e 2011 (GALLINA, 2011a). Para este trabalho, foram escolhidas 47 propriedades, que juntas representam 93,36% das ocorrências de propriedades temporais de acordo com os dados anotados. A Tabela 4.3 lista essas propriedades e o tipo de tempo ao qual estão associadas no conjunto de dados da DBPedia.

Tabela 4.3 Propriedades temporais suportadas

Propriedade	**Tipo**	**Propriedade**	**Tipo**
releaseDate	Data	publicationDate	Data
birthDate	Data	serviceEndYear	Ano
activeYearsStartYear	Ano	serviceStartYear	Ano
deathDate	Data	debut	Data
activeYearsStartDate	Data	buildingStartDate	Data
activeYearsEndDate	Data	modelStartYear	Ano
activeYearsEndYear	Ano	functionStartYear	Ano
formationYear	Ano	extinctionDate	Data
added	Data	closingDate	Data
deathYear	Ano	modelEndYear	Ano
discovered	Data	landingDate	Data
foundingYear	Ano	beatifiedDate	Data
birthYear	Ano	canonizedDate	Data
completionDate	Data	electionDate	Data
recordDate	Data	buildingEndDate	Data
openingDate	Data	endDate	Data
formationDate	Data	productionStartDate	Data
productionStartYear	Ano	retired	Data
date	Data	productionEndDate	Data
productionEndYear	Ano	introduced	Data
startDate	Data	freeFlightTime	Duração
foundingDate	Data	timeInSpace	Duração
extictionYear	Ano	runtime	Duração
undraftedYear	Ano		

Pode-se perceber que existem três tipos de propriedades: data, ano e duração. Também é possível notar que diversas propriedades poderiam ser agrupadas, pois aparecem duas vezes com tipos diferentes. Logo, classificando as propriedades de

acordo com os tipos de entidade de tempo associadas a elas, existem propriedades de quatro modalidades: data, ano, data e ano, e duração. As datas são armazenadas no conjunto de dados obedecendo ao formato YYYY-MM-DD, os anos são armazenados no formato YYYY e as durações como um número real (*double*), que representa o número de segundos.

4.5 Módulos de Funcionamento

Essa seção detalha todos os módulos sequenciais que são executados para que o processo da busca temporal seja realizado. Os módulos são: identificação da pergunta, extração e normalização dos dados contidos na pergunta, montagem e execução da consulta SPARQL e o tratamento do retorno e exibição dos resultados. Cada módulo é apresentado definindo seu funcionamento básico, quais as variações para cada tipo de pergunta e variação de entradas, quais os problemas enfrentados e soluções empregadas para seu funcionamento. Por fim, há um resumo geral sobre o processo como um todo.

A subseção 4.5.1 trata do módulo de identificação da pergunta. A subseção 4.5.2 define todo o processo de extração e normalização dos dados existentes na própria pergunta. Na subseção 4.5.3, é explicada a montagem da consulta a partir dos dados obtidos pelos dois processos anteriores. A subseção 4.5.4 mostra como o retorno da consulta é lido, tratado e exibido de forma conveniente para o usuário. Por fim, na subseção 4.5.5 há um resumo geral sobre o funcionamento dos módulos em conjunto.

4.5.1 Identificação da Pergunta

A primeira etapa do processo realizado pela ferramenta é ler a pergunta submetida pelo usuário e analisa-la sintaticamente para descobrir se é uma pergunta válida, e, em caso afirmativo, de qual dos tipos de perguntas se trata. A Figura 4.5 ilustra, com algumas simplificações, como o texto da pergunta é tratado para a identificação da pergunta. Nesse momento, a preocupação não é analisar a semântica nem tratar as informações, mas apenas validar a entrada e identificar qual o caso de pergunta envolvido.

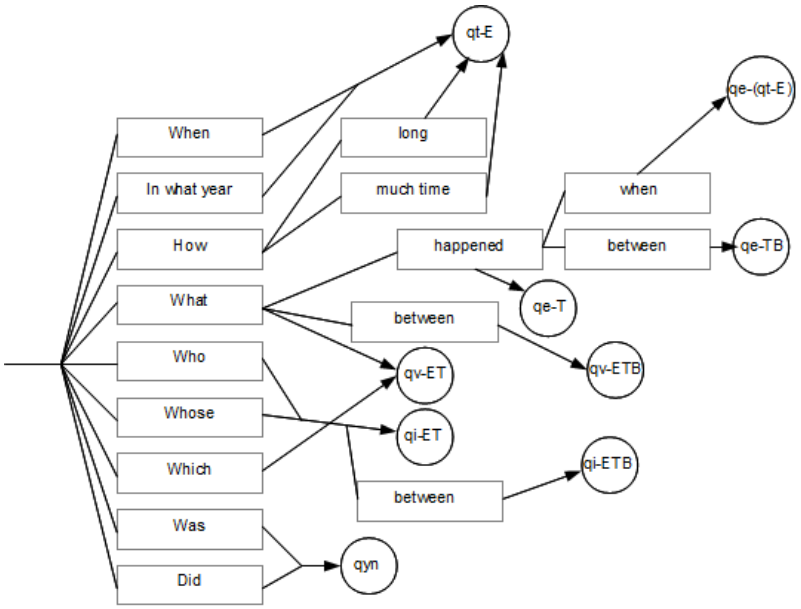

Figura 4.5 Fluxograma de identificação de perguntas

4.5.2 Extração e Normalização dos Dados

Sabendo qual o tipo de pergunta que o usuário digitou, é possível determinar, conforme mostrado na Tabela 4.1, quais os dados que foram fornecidos na própria pergunta. Esses dados são, então, identificados, separados e extraídos do texto original e guardados em variáveis individuais. Para a integração entre os módulos funcionar corretamente, esses dados precisam ser normalizados para formatos que podem ser devidamente reconhecidos pelo serviço SPARQL, que tratará a consulta nos módulos subsequentes.

A extração é bastante trivial: como mostrado na Tabela 4.2 e na Figura 4.5, as palavras-chave que definem a sintaxe das perguntas são bem definidas e não são ambíguas. A identificação da pergunta pode ser feita através de um algoritmo simples que vai, a cada palavra lida, mudando o estado atual até chegar no tipo de pergunta específico analisado e finalmente, para cada tipo de pergunta, a ordem de ocorrência dos dados fornecidos na frase está também bem definida. Há um número fixo de predicados suportados, todos podendo ser reconhecidos na frase. Os predicados agem também como separadores, ou seja, o que vem antes deles é com certeza o sujeito e o que vem após é o tempo, facilitando o processo de separação dos dados fornecidos.

Sobre o tratamento desses dados, segue um detalhamento sobre as normalizações efetuadas sobre cada variável possível: sujeito, predicado e tempo. Em seguida, são citados quais tipos de normalização que são usados para cada pergunta suportada.

Sujeito

O sujeito lido da pergunta pode ser simples ou composto. Caso o sujeito seja composto, os espaços em branco que separam suas palavras componentes são convertidos para *underscores* "_", para ficar de acordo com o padrão adotado pela DBPedia. Outra modificação necessária é garantir que cada palavra componente comece por uma letra maiúscula e seja seguida por letras minúsculas.

Sujeitos são recursos definidos por uma URI no arquivo de dados interligados RDF, portanto cada um deles deve ser precedido de "resource:", que é o prefixo utilizado para "http://dbpedia.org/resource/".

Como um dos objetivos é garantir uma alta eficiência nos resultados da busca, são previstos os casos em que o usuário não sabe o nome completo ou erra algum dos termos. Por isso, são criadas instâncias de sujeito de todos os prefixos ou sufixos do sujeito original, e essas instâncias são salvas em um vetor ordenado pelo número de palavras de forma descendente. O motivo de ordenar as instâncias de sujeito dessa maneira é explicado com mais detalhe na próxima subseção. Segue um exemplo de normalização de sujeito:

Sujeito original: *Howard james HUBBARD*
Sujeito adaptado: *resource:Howard_James_Hubbard*
Vetor de instâncias de sujeito:
{resource:Howard_James_Hubbard,
resource:Howard_James,
resource:James_Hubbard,
resource:Howard,
resource:James,
resource:Hubbard}

Predicado

Para a normalização do predicado ficar em um formato compatível para a consulta, é criada uma tabela de correspondência, onde associa-se a forma verbal do predicado base presente na pergunta ao predicado formal da DBPedia, que consta na Tabela 4.3. Considerando a simplicidade do idioma inglês, na maioria dos casos essa associação é bastante simples. Por exemplo, a ocorrência de "*was born*" está associada ao predicado formal "*birth*". Há alguns predicados que não variam, como "*start*" que na DBPedia é também representado por "*start*". Alguns predicados são irregulares, como é o caso de "*became inactive*", que é representado por "*activeYearsEnd*".

A segunda parte da normalização é completar o predicado base obtido pela tabela de correspondência com o seu sufixo. A coluna Tipo listada na Tabela 4.3 define o tipo de data que é representado pelos predicados. Dependendo de cada predicado, o sufixo pode ser "*Date*", significando uma data, "*Year*", significando um ano, vazio (ou seja, o próprio predicado-base já é o predicado final) , significando uma data, ou especial, significando uma duração. No caso de definir uma duração, também não é necessário sufixo, pois a tabela apresentada já contempla o predicado final.

Para garantir a eficiência e alcançabilidade da solução, são tomadas algumas medidas:

- Caso a pergunta peça por um ano e haja um predicado "*Year*", esse é o utilizado para montar o predicado final na consulta.
- Caso a pergunta peça um ano, mas só haja predicado "*Date*", utiliza-se esse predicado. Mesmo não sendo o que foi originalmente pedido, pode-se pegar o ano da data completa e comparar com o ano solicitado.
- Caso a pergunta peça uma data, só é possível retornar um predicado "*Date*", e caso não haja não será possível fazer a consulta, pois o usuário queria algo mais restrito do que o conteúdo disponível dos dados.
- Quando o predicado possui os dois tipos, data e ano, se for solicitado um ano na pergunta, os dois são utilizados, mas com prioridade ao predicado do tipo "*Year*". Se for solicitada uma data, só é utilizado o predicado "*Date*".

Por fim, é feita uma concatenação entre "dbpedia:", que é um sinônimo para "http://dbpedia.org/ontology/", e o predicado obtido para deixa-lo na forma normal tal como deve aparecer na consulta.

Exemplo:

Pergunta: *When did Abraham Lincoln die?*

Predicado: *die*

Predicado-base para die: *death*

Tipos para death: *data e ano*

Pergunta quis data ou ano: *data*

Predicado final: *deathDate*

Tempo

O tempo pode ocorrer de três formas:

- pode ser uma data completa no padrão americano, separada por barras, ou seja, no formato MM/DD/YYYY;
- pode ser um ano, representado como um número inteiro de 4 dígitos YYYY; ou
- pode ser um período que abrange tudo que está entre o tempo 1 e o tempo 2, onde esses tempos podem ser de qualquer um dos tipos acima e vêm seguidos de um *between* e separados por *in* (*between t1 and t2*)

A normalização desses tempos consiste em transformar o formato lido no formato da DBPedia. Se o tempo for um ano, não é necessário fazer nada, e se for uma data, esta é convertido para o formato YYYY-MM-DD.

4.5.3 Montagem e Execução da Consulta SPARQL

O módulo responsável pelas consultas SPARQL é a parte mais importante do trabalho, pois é onde são unidos os dados lidos na pergunta e as variáveis buscadas para respondê-la. As consultas, cuja sintaxe é semelhante à do SQL, são feitas sobre o grafo RDF gerado a partir do conjunto de dados com as triplas em RDF. Seguem as definições e comentários sobre como é montada a consulta para cada tipo de pergunta.

Q_T-E:

PREFIX dbpedia: <http://dbpedia.org/ontology/>
PREFIX resource: <http://dbpedia.org/resource/>
SELECT ?data
WHERE { { sujeito[0] predicado ?data} UNION { sujeito[1] predicado ?data} ... sujeito[n] predicado ?data} }

A única variável da consulta é a data. São utilizados os sujeitos e o predicado já normalizados no passo anterior. O motivo pelo qual todas as combinações de prefixos e sufixos de sujeitos foram geradas, e o motivo pelo qual o vetor de sujeitos foi ordenado do maior para o menor são, respectivamente, para maximizar o número de resultados obtidos para o sujeito consultado e para garantir que, quanto mais semelhantes os resultados forem com a busca realizada, mais no topo eles se encontram. Quando o sujeito possui apenas uma palavra, não é feita a união e a consulta é feita apenas com a respectiva instância.

Para os próximos casos, a palavra-chave PREFIXOS substitui as duas primeiras linhas da consulta, visto que elas se repetem em todas as consultas.

Q_E-T:

PREFIXOS
SELECT DISTINCT ?sujeito ?predicado
WHERE { ?sujeito ?predicado data^^<http://www.w3.org/2001/XMLSchema#date> }

Ao contrário do Q_T-E, a única constante é a data. O sujeito e o predicado são variáveis que devem estar associados àquela data. Quando a pesquisa é feita por ano, o

ideal seria retornar também aquelas triplas que se referem àquele ano, mas que estão representadas como uma data. Como o predicado é uma variável, não se sabe durante a consulta quais predicados têm somente ano e quais têm datas completas em sua estrutura. Nesse caso, a consulta é feita da seguinte maneira. Não confundir a variável data (?data) que representa a data completa do ano em questão, e a constante data (data) que representa o ano proveniente da pergunta.

```
PREFIXOS
SELECT DISTINCT ?sujeito ?predicado ?data
WHERE { { ?sujeito ?predicado data^^<http://www.w3.org/2001/XMLSchema#gYear> }
UNION { ?sujeito ?predicado ?data
FILTER (?data) >= “data-01-01”^^<http://www.w3.org/2001/XMLSchema#date> &&
(?data) <= “data-12-31”^^<http://www.w3.org/2001/XMLSchema#date> } }
```

As linhas do resultado contêm três campos nesse caso: o sujeito, o predicado e a data, sendo que, no topo, aparecem as datas que são exatamente iguais ao ano fornecido, e em seguida aparecem os eventos guardados em forma de data completa que dizem respeito ao mesmo ano, entre 01 de janeiro e 31 de dezembro.

Q_I-ET e Q_V-ET:

```
PREFIXOS
SELECT DISTINCT ?sujeito ?data
WHERE { ?sujeito predicado data^^<http://www.w3.org/2001/XMLSchema#gYear> }
UNION { ?sujeito predicado ?data
FILTER (?data) >= “data-01-01”^^<http://www.w3.org/2001/XMLSchema#date> &&
(?data) <= “data-12-31”^^<http://www.w3.org/2001/XMLSchema#date> } }
```

A consulta correspondente à Q_I-ET e à Q_V-ET é a mesma, pois os mesmos elementos são sabidos e procurados. As únicas diferenças são semânticas e variam apenas no modo como a pergunta deve ser feita e respondida. A lógica disso está toda implícita na ontologia. Por exemplo, um predicado relativo a uma pessoa, como *birthDate*, está associado somente a pessoas no conjunto de dados. Assim, a consulta executada retornará sujeitos que definem pessoas.

Pode-se perceber que essa consulta é bastante similar à da Q_E-T. Neste caso, a consulta citada já é um exemplo do caso especial mencionado na segunda consulta da Q_R-T. O caso normal, de pesquisa por data fixa, é trivial, bastando utilizar o esquema *date* sem a união.

Uma diferença em relação ao Q_E-T é que, enquanto no caso anterior não se conhecia o predicado, nesse caso é sabido com antecedência se é um predicado que também possui equivalente com a data completa na ontologia ou somente com o ano. Assim, essa consulta especial só é realizada quando efetivamente há chance de produzir resultados dessa maneira.

Q_{YN}:

```
PREFIXOS
SELECT “yes”
WHERE { sujeito predicado data^^<http://www.w3.org/2001/XMLSchema#date> }
```

Essa é a consulta mais simples de todas, que consiste em verificar se a tripla correspondente à associação entre o sujeito, o predicado e a data fornecidos faz parte do grafo RDF. Exatamente como no caso anterior, nas pesquisas por ano sobre um predicado com ano e data, é feita a união entre os resultados dos dois casos.

É importante citar que o SPARQL possui uma cláusula especial ASK, que tem essa mesma utilidade de apenas verificar se uma tripla pertence ao grafo. Essa palavra reservada substitui o SELECT na consulta. Essa cláusula não foi utilizada porque a resposta escolhida para o retorno não foi Sim/Não, mas Sim/Não/Não sei. Sendo assim, foi necessário consultar dessa maneira. O "não sei" funciona assim: caso não haja uma equivalência para a tripla (sujeito, predicado, data) fornecida pelo usuário, tenta-se uma equivalência para o par (sujeito, predicado). Deste modo, se existe esse par com outra data, a resposta é de fato "não" e, caso contrário, nada pode ser afirmado sobre o par.

Q_E-(Q_T-E):

A montagem dessa consulta é feita em duas etapas. Primeiro, é resolvido o Q_T-E interno. O retorno dessa consulta é utilizado diretamente como entrada para resolver o Q_E-T externo, assim como em uma função composta. Essas duas consultas já foram explicadas anteriormente.

Q_E-T^B:

```
PREFIXOS
SELECT DISTINCT ?sujeito ?predicado ?data
WHERE { ?sujeito ?predicado ?data
FILTER ( (?data) >= "data1"^^<http://www.w3.org/2001/XMLSchema#date> &&
(?data) <= "data2"^^<http://www.w3.org/2001/XMLSchema#date>  }
ORDER BY ASC(?data)
```

A diferença em relação à Q_E-T é que sempre é pesquisado um período entre duas datas completas, por questões de simplicidade e pelo efeito colateral disso ser muito pequeno. Assim, uma minoria de predicados, que são os que não possuem uma representação de data completa, e só aparecem associados ao ano, não podem ser pesquisados nesse tipo de pergunta.

As datas fornecidas como anos são sempre convertidas para o primeiro ou último dias daquele ano. Sendo assim, uma consulta de 1990 a 05 de março de 2005, por exemplo, irá pesquisar as datas entre 01 de janeiro de 1990 e 05 de março de 2005. Como esse tipo de consulta representa um intervalo definido de tempo, que pode conter muitos eventos, as datas são ordenadas por data de forma crescente.

Q_I-ET^B e Q_I-ET^B:

```
PREFIXOS
SELECT DISTINCT ?sujeito ?data
WHERE { ?sujeito predicado ?data
FILTER ( (?data) >= "data1"^^<http://www.w3.org/2001/XMLSchema#date> &&
(?data) <= "data2"^^<http://www.w3.org/2001/XMLSchema#date>  }
ORDER BY ASC(?data)
```

A diferença entre a Q_I-ET^B e a Q_E-T^B é a mesma que entre a Q_I-ET e a Q_E-T: a restrição por predicado. A diferença é que neste caso, como sempre é utilizada a data, o esquema utilizado não varia.

4.5.4 Tratamento do Retorno e Exibição

As consultas montadas conforme definido na seção anterior são submetidas ao serviço HTTP do Joseki, *temporal*, via PHP. Para isso, é utilizada a biblioteca

SimpleXML, que já vem integrada ao PHP. Essa biblioteca permite ler, criar e manipular arquivos XML de maneira simples e poderosa.

É possível carregar um arquivo local ou uma URL que representa um arquivo XML. A segunda opção é a utilizada pela ferramenta. Com um único comando, é feita a chamada ao serviço, que roda a consulta SPARQL e devolve os resultados em um arquivo XML, que é lido e carregado em uma variável local do PHP, de acordo com o trecho de código abaixo:

```
$url = "http://localhost:2020/temporal?query=
".urlencode($consulta)."&output=xml&stylesheet=%2Fxml-to-html.xsl";

$xml = simplexml_load_file($url);
```

A Figura 4.5 ilustra o formato de um arquivo XML retornado pelo SPARQL.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="/xml-to-html.xsl"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
 <head>
  <variable name="sujeito"/>
  <variable name="predicado"/>
  <variable name="data"/>
 </head>
 <results>
  <result>
   <binding name="sujeito">
    <uri>http://dbpedia.org/resource/The_Decalogue</uri>
   </binding>
   <binding name="predicado">
    <uri>http://dbpedia.org/ontology/releaseDate</uri>
   </binding>
  </result>
  <result>
   <binding name="sujeito">
    <uri>http://dbpedia.org/resource/Bony_Wilfried</uri>
   </binding>
   <binding name="predicado">
    <uri>http://dbpedia.org/ontology/birthDate</uri>
   </binding>
  </result>
 </results>
</sparql>
```

Figura 4.5 Retorno da consulta SPARQL em XML

Para efetuar a leitura dos campos de interesse para responder à pergunta, especificamente *results/result/binding/uri* e *results/result/binding/literal*, foi utilizada a função SimpleXMLElement::xpath, que pesquisa o elemento dado pela expressão

XPath que recebe como parâmetro. A função se mostrou bastante útil já que os caminhos dos resultados de interesse no XML, embora longos e com atributos, seguem sempre o mesmo padrão.

Os resultados da consulta são lidos para uma matriz cujas linhas representam o i-ésimo resultado da consulta e as colunas representam o que está sendo representado (sujeito, predicado, ou tempo).

O passo final é utilizar essa matriz para montar as respostas da aplicação em uma linguagem que o usuário final entenda, respondendo o que ele perguntou inicialmente. As respostas são montadas no formado Sujeito + Predicado + Tempo, uma vez que após a consulta, seja qual for o tipo de pergunta, todos esses dados já estão disponíveis. Caso isso não aconteça, o motivo é o XML ter vindo com a sub-árvore *results* vazia. Neste caso, será respondido ao usuário que não é possível responder à pergunta dele com os dados disponíveis. Quando o tempo é óbvio, em perguntas por uma data específica, o tempo é ocultado da resposta.

Além disso, os predicados são transformados para a forma verbal de resposta através de uma tabela de correspondência que faz algo muito semelhante ao processo inverso da primeira transformação. As únicas diferenças se devem a ajustes devidos a detalhes sobre formas verbais em inglês: o verbo auxiliar "do", por exemplo, aparece na forma interrogativa, mas não na afirmativa, por isso é eliminado. Desse modo, um predicado como "*deathYear*", por exemplo, é transformado para a forma verbal "*died*" na resposta. Igualmente, a data sofre o processo inverso para voltar ao formato conhecido pelo usuário. A Tabela 4.4 demonstra como a resposta é montada para cada tipo de pergunta.

Tabela 4.4 Formato das respostas temporais

Sigla	Resposta
Q_T-E	[sujeito] (was) [predicado] (on [data])
Q_E-T	[sujeito] (was) [predicado] in/on [data]
Q_I-ET	[sujeito] / [sujeito] (was) [predicado] on [data]
Q_V-ET	[sujeito] / [sujeito] (was) [predicado] on [data]
Q_{YN}	Yes/No/We don't know
Q_E-(Q_T-E)	[sujeito] (was) [predicado] in/on [data]
Q_E-T^B	[sujeito] (was) [predicado] in/on [data]
Q_I-ET^B	[sujeito] / [sujeito] (was) [predicado] on [data]
Q_V-ET^B	[sujeito] / [sujeito] (was) [predicado] on [data]

4.6 Apresentação da Ferramenta

A ferramenta desenvolvida funciona em qualquer navegador e tem a tela inicial parecida com a dos motores de busca convencionais. Há um campo para submissão da pergunta e dois botões: um para enviar a pergunta e outro para exibir a ajuda. Os resultados aparecem em forma de frases que respondem à pergunta, separados por quebras de linha.

A seguir, são apresentadas algumas capturas de tela mostrando o funcionamento da aplicação, através de alguns casos de teste. A Figura 4.6 apresenta a tela inicial da ferramenta, onde se espera que uma pergunta seja feita, e a Figura 4.7 mostra uma tela em que são exibidos os resultados referentes a uma pergunta.

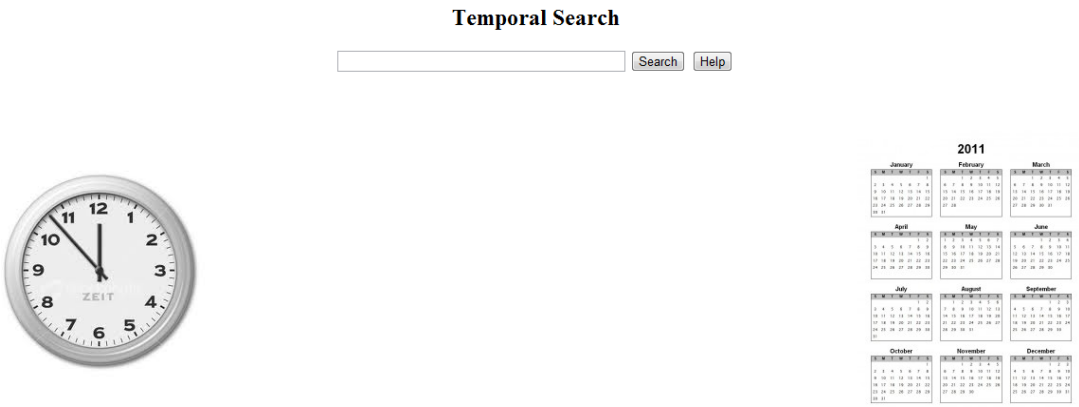

Figura 4.6. Apresentação da ferramenta

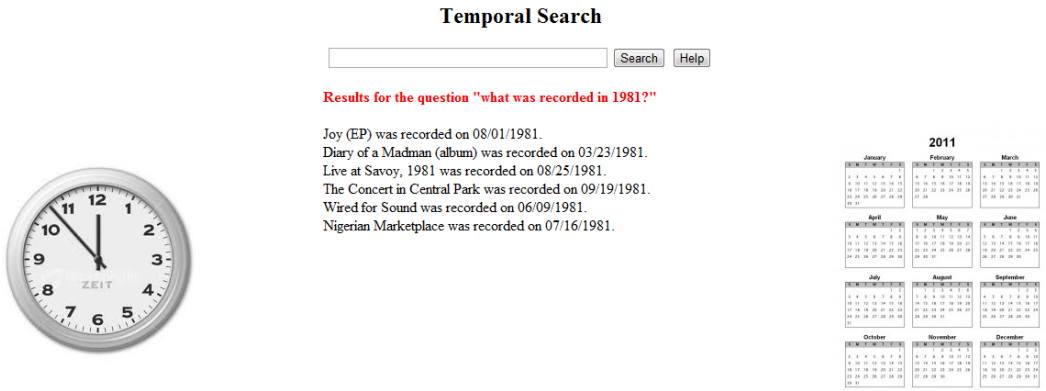

Figura 4.7 Exibição dos resultados de uma pergunta

A Figura 4.8 corresponde ao resultado de uma pergunta Q_E-T, "*What happened in 1782?*". Nessa consulta, o usuário está interessado em saber todos os eventos que aconteceram em 1782, ou seja, todas as triplas que representam os eventos associados a predicados ano com o terceiro termo da tripla igual a 1782 e eventos associados a predicados data com o terceiro termo da tripla na forma 1782-MM-DD. No topo dos resultados, aparecem as correspondências exatas, onde foi salvo apenas o ano. Já que a busca também é feita pelo ano, há mais chances do resultado procurado aparecer entre essas linhas. No final, são incluídos também os eventos com datas completas do mesmo ano, para garantir mais opções de resposta à questão feita.

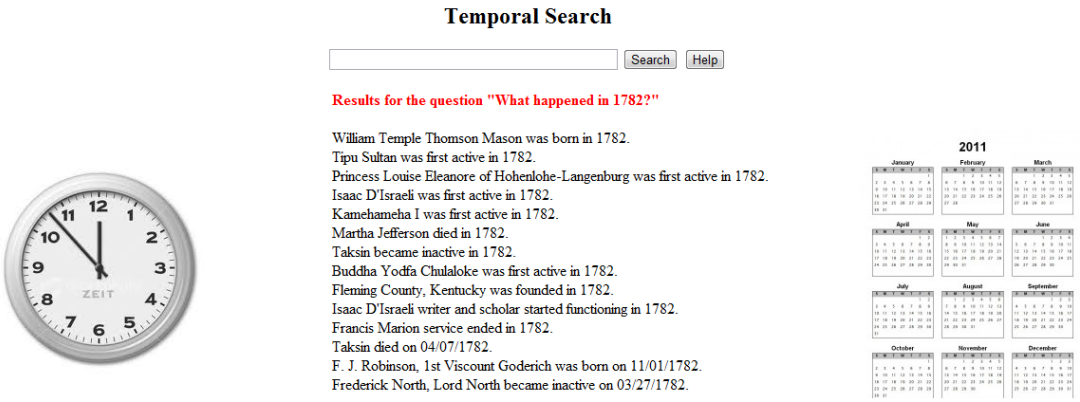

Figura 4.8 Respostas a uma pergunta Q_E-T com ano

A pergunta que aparece na figura 4.9 é uma das que seriam feitas com mais frequência no uso cotidiano. Deseja-se saber quem nasceu na primeira semana de julho de 1952. Essa é uma pergunta Q_I-ET^B, onde são procurados sujeitos ligados ao predicado de nascimento e com datas dentro do intervalo especificado. Essas respostas, que dependendo das datas em questão, do tamanho do conjunto de dados e do intervalo de tempo podem aparecer em grande número, são ordenadas da mais antiga para a mais recente.

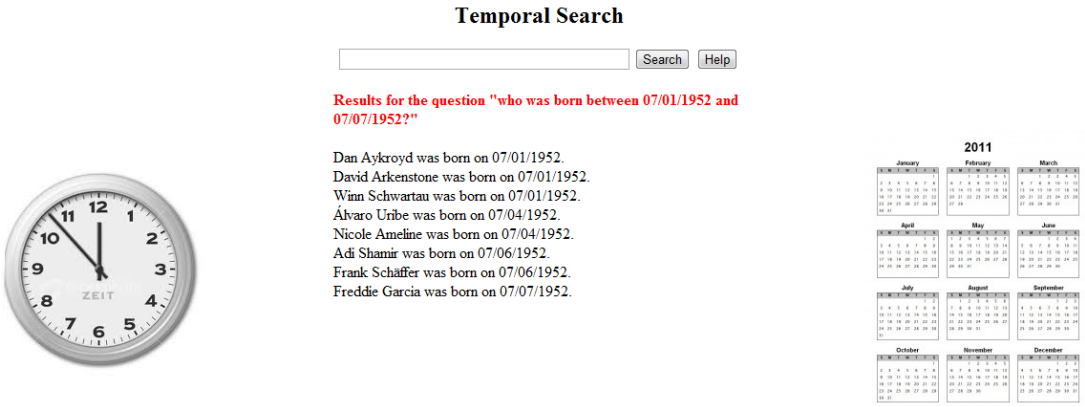

Figura 4.9 Respostas a uma pergunta Q_I-ET^B

Por fim, uma consulta feita propositalmente sobre um evento relacionado a um sujeito que não existe nos dados armazenados. É feita uma consulta e não é encontrada uma tripla que represente a morte do personagem em 10 de novembro de 1990. Então, o sistema pesquisa a existência de qualquer tripla sobre a morte desse personagem, e como não é encontrada nenhuma, a resposta dada não é nem sim e nem não. Responde-se “não sabemos”. A figura 4.10 é uma captura de tela desse caso:

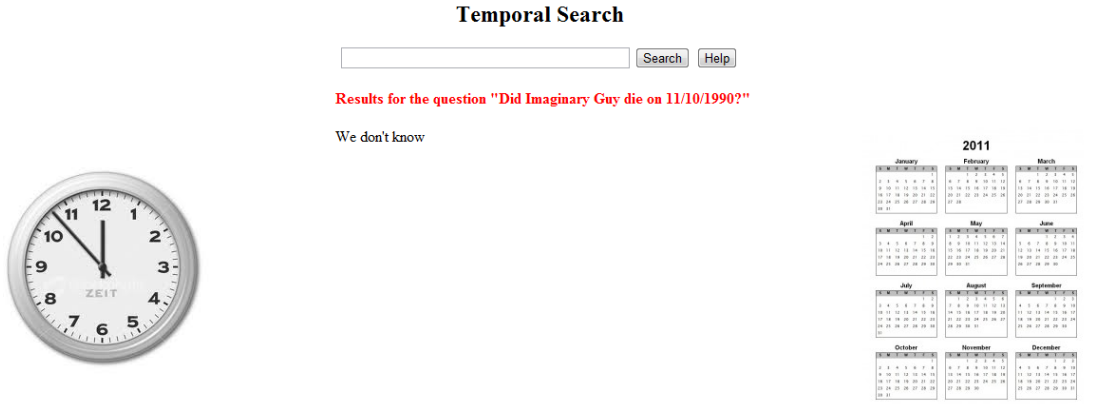

Figura 4.10 Resposta a uma pergunta Q_{YN} sem dados associados

4.7 Resumo Geral

A Seção 4 apresentou a ferramenta implementada para a resposta de perguntas com relevância temporal. Foram explicados com detalhes todos os módulos envolvidos no processo de funcionamento da ferramenta. Foram detalhados quais os requisitos e configurações técnicas necessárias para a utilização da ferramenta e foram mostrados diversos exemplos de casos de funcionamento através de capturas obtidas diretamente da ferramenta em ação.

As perguntas são lidas e processadas para averiguar que tipo de pergunta foi feita e se foi uma pergunta válida. Em caso afirmativo, os dados são extraídos da pergunta e normalizados para um formato que é compatível com as triplas de dados interligados RDF. Essas informações, que são obtidas a partir da própria pergunta, são utilizadas para a construção da consulta SPARQL, que retorna um conjunto de triplas que satisfazem às restrições definidas pelo tipo de pergunta em questão. A saída é processada e retornada para o usuário em linguagem natural, de modo que é retornado o maior número possível de informações que possam responder ao que foi perguntado e que as respostas que melhor respondem à pergunta aparecem primeiro.

A principal contribuição dessa ferramenta é: uma alternativa para busca por palavras-chave especificamente sobre dados temporais extraídos de páginas e armazenados em formato de triplas de dados interligados que constituem um grafo RDF, de modo que conjuntos de dados que obedecem à ontologia da DBPedia podem ser pesquisados facilmente na linguagem natural da língua inglesa através de perguntas que se relacionam ao tempo. Isso ocorre com um número consideravelmente pequeno de restrições quanto à sintaxe utilizada na pergunta, sendo o sistema claro e direto, pois entende a linguagem nativa do usuário.

5 EXPERIMENTOS

Os experimentos realizados durante este trabalho tiveram como objetivo medir a qualidade dos resultados gerados pela ferramenta desenvolvida, no que diz respeito ao quão bem a ferramenta responde a perguntas feitas em linguagem natural e com algum valor temporal. Como não se tinha o conhecimento, até o momento da publicação deste trabalho, de nenhuma ferramenta suficientemente similar para que pudesse ser feita uma comparação de qualidade para os mesmos casos de teste (isto é, ser usada como *baseline*), decidiu-se elaborar manualmente gabaritos esperados para cada um desses casos, e compará-los com os resultados obtidos ao realizar esses testes na ferramenta.

Este capítulo subdivide-se em três seções. Na seção 5.1, são descritos os dados referentes à forma como foram feitos os experimentos. A seção 5.2 contém detalhes específicos sobre cada caso de teste realizado, os resultados obtidos e a análise sobre esses resultados. A seção 5.3 finaliza o capítulo fazendo uma análise geral dos resultados obtidos com os experimentos feitos sobre a ferramenta.

5.1 Configuração dos Experimentos

Os testes foram escolhidos de modo a evidenciar os pontos fortes e fracos da ferramenta no que diz respeito a responder a perguntas temporais, bem como investigar o quanto é possível responder corretamente através dessa ferramenta. Foram calculadas as taxas de precisão e a revocação para cada um dos casos, para evidenciar os resultados obtidos.

As taxas de precisão e revocação (BAEZA-YATES, 1999) foram calculadas da seguinte forma: inicialmente, foram anotadas as respostas esperadas para cada grupo de perguntas que compôs cada um dos casos de teste computado, representando o total de respostas esperadas no teste. Em seguida, foram anotadas as respostas retornadas para cada uma das perguntas feitas, representando o total de respostas encontradas pelo teste. As respostas encontradas foram subdivididas em acertos, para respostas que eram esperadas, e erros, para respostas que não eram esperadas.

- A precisão foi obtida através da razão entre o número de acertos e o número de respostas encontradas, portanto, ela mede a porcentagem de acertos dentre as respostas obtidas pela ferramenta.
- A revocação foi obtida através da razão entre o número de acertos e o número de respostas esperadas, portanto, ela mede a porcentagem de respostas esperadas que a ferramenta conseguiu obter.

A base de dados utilizada para os experimentos foi o mesmo subconjunto do conjunto de dados da DBPedia citado na seção 4.2, *Raw Infobox Properties*. O conjunto

de dados utilizado tinha um tamanho de 134MB e 1.010.882 triplas. Os dados representam, em sua maioria, dados da versão em inglês da Wikipedia. Foram realizadas um total de 260 perguntas divididas em quatro casos de testes distintos para avaliar a ferramenta em diferentes aspectos. Foram avaliadas todas as respostas retornadas pela ferramenta, razão pela qual o número de respostas esperadas e encontradas é bem maior do que o número de perguntas para algumas modalidades de pergunta, o que também explica porque o número total de perguntas de cada caso de teste não foi uniforme. No último caso de testes, os resultados esperados foram anotados diretamente da Wikipedia, conforme é explicado na seção 5.2.

Não foi possível realizar experimentos sobre as triplas geradas a partir do trabalho de (GALLINA, 2011a), o qual foi a motivação deste trabalho, devido ao referido trabalho ainda estar em andamento no momento desta publicação. Porém, como explicado ao longo desse texto, a ferramenta funciona para qualquer conjunto de dados em RDF que siga a ontologia da DBPedia, como é o caso do trabalho mencionado. Isso reafirma a escalabilidade e reusabilidade dos dados interligados.

5.2 Execução dos Testes

Esta seção se destina a descrever detalhadamente os quatro casos de teste realizados, informando mais detalhes sobre as perguntas efetuadas e mostrando os resultados obtidos e a respectiva interpretação dos resultados.

No primeiro caso de testes, foram realizadas 20 perguntas dos tipos Q_E-T e Q_E-T^B, das quais 9 foram feitas sobre um ano, 3 sobre uma data completa, 5 sobre um período entre anos e 3 sobre um período entre datas. Essa distribuição foi escolhida de forma arbitrária. A Tabela 5.1 mostra os resultados obtidos.

Tabela 5.1 Primeiro caso de testes

Total	Encontradas	Acertos	Erros	Precisão	Revocação
311	311	311	0	100%	100%

A Tabela 5.1 mostra que a qualidade da ferramenta para esses tipos de perguntas feitas, respeitando a sintaxe, sobre dados contidos no conjunto de dados utilizado, é total. O resultado foi conforme o esperado, já que esses dados estão representados no conjunto de dados e tudo que foi feito foi um filtro pelo tempo.

O segundo caso de testes consistiu em realizar 100 perguntas Q_{YN} e Q_T-E, das quais 65 foram feitas sobre datas completas e 35 sobre anos. Além disso, 93 tinham sujeito composto e 7 tinham sujeito simples. Essas características foram escolhidas apenas para refletir a distribuição dos dados, sem nenhum motivo especial. A Tabela 5.2 mostra os resultados obtidos:

Tabela 5.2 Segundo caso de testes

Total	Encontradas	Acertos	Erros	Precisão	Revocação
100	100	96	4	96%	96%

A Tabela 5.2 mostra que houve alguns erros. Apesar das perguntas terem sido feitas conforme a sintaxe esperada e sobre dados devidamente representados no conjunto de dados utilizado, alguns pares de sujeitos desse conjunto de dados que são prefixos um do outro causaram esses problemas. Como a ferramenta tenta pesquisar com todas as combinações de prefixos e sufixos de sujeitos compostos, pode haver resultados errados nos casos em que o sujeito prefixo do pesquisado aparece no conjunto de dados associado a uma data diferente da data associada ao sujeito pesquisado. Por exemplo, na pergunta "*When was Morton Downey Jr born?*", Morton Downey aparece no conjunto de dados com a data 11/14/1901, mas o sujeito pesquisado possui a data 12/09/1932 e a ferramenta responde a primeira.

No terceiro caso de testes, foram feitas 15 perguntas Q_E-(Q_T-E), das quais 11 tinham o tempo que responde ao Q_T-E interno representado como datas completas e 4 como anos. A Tabela 5.3 mostra os resultados obtidos:

Tabela 5.3 Terceiro caso de testes

Total	Encontradas	Acertos	Erros	Precisão	Revocação
164	145	145	0	100%	88,4%

Pode-se perceber na Tabela 5.3 que, da mesma forma que no primeiro caso, a precisão foi total por se tratar de um filtro de tempo. Isso significa que todas as respostas encontradas estavam associadas ao tempo esperado, que é o tempo que responde à pergunta Q_T-E interna. No entanto, a revocação foi de 88,4%. Isso se deve ao fato de que algumas respostas esperadas não foram retornadas pela ferramenta. As situações que causaram isso foram quando a resposta do q_T-E interno é restritiva demais para responder ao Q_T-E externo. Ao perguntar o que aconteceu durante o tempo de um evento que está representado como uma data completa, são retornados apenas eventos ligados à mesma data. Mas isso é transparente para o usuário, que não sabe como a representação interna é feita e está sempre esperando que as respostas digam respeito também a datas próximas à da pergunta interna. Esse é um efeito colateral da transparência dada ao usuário que não traz tantos prejuízos, já que essas respostas não encontradas podem ser obtidas fazendo-se as duas perguntas em sequência: primeiro faz-se um Q_T-E idêntico à pergunta interna do Q_E-(Q_T-E) e, caso a resposta seja uma data completa mas deseja-se eventos em uma faixa maior que a data, pode-se fazer um Q_E-T^B utilizando uma faixa de datas que contenha a data obtida anteriormente.

O quarto caso de testes se destinou a comparar os resultados obtidos com o que seria esperado encontrar na Wikipedia, visto que, essencialmente, DBPedia é construída a partir dela. Para isso, foram feitas perguntas temporais em linguagem natural que foram encontradas arbitrariamente na Web. Para evitar perdas devido ao uso de somente uma parte do conjunto de dados da DBPedia, todas as perguntas feitas diziam respeito a sujeitos que faziam parte do conjunto de dados mencionado na seção 5.1. O motivo desse caso de teste ter sido feito dessa maneira foi para avaliar a qualidade da

ferramenta de uma forma mais genérica, da forma que seria utilizada para pesquisar dados da Wikipedia em forma de perguntas temporais. Esse teste foi importante também porque os experimentos realizados sobre o conjunto de dados utilizado pela DBPedia sempre traziam resultados muitos bons, razão pela qual não foram mostrados mais casos de teste, já que praticamente não havia erros.

No quarto caso de testes, foram feitas 125 perguntas no total, as quais se subdividiram em perguntas Q_{YN} (55 perguntas), Q_T-E (45 perguntas) e Q_I-ET (25 perguntas). Os resultados obtidos constam na Tabela 5.4:

Tabela 5.4 Quarto caso de testes

Total	Encontradas	Acertos	Erros	Precisão	Revocação
125	96	96	0	100%	76,8%

Os motivos pelos quais a revocação obtida a partir dos testes cujas estatísticas estão representadas na Tabela 5.4 foi mais baixa são: (i) perguntas com sinônimos e formatos não suportados pela ferramenta: apesar da ferramenta desenvolvida não prever esses casos, o objetivo desse teste era justamente avaliar qual o preço que é pago, em perda de qualidade, em função dessas simplificações; e (ii) perguntas sobre predicados temporais que não são utilizados pela ferramenta, seja porque a DBPedia não os guarda a partir das informações da Wikipedia corretamente, porque a ferramenta não utiliza todos os predicados temporais da DBPedia, ou porque as palavras utilizadas para referenciar o predicado não puderam ser convertidas no predicado da ontologia da DBPedia por não estarem na tabela de conversão. A precisão foi de 100% pois todos os resultados encontrados eram conforme os esperados, o que prova que a qualidade da ferramenta é bem alta para dados corretamente representados. Acredita-se que os resultados foram satisfatórios devido ao fato das perguntas terem sido feitas em um formato bastante natural sobre dados da Web, e mesmo assim, 3 em cada 4 perguntas foram respondidas com sucesso, o que se deve, em grande parte, ao bom trabalho do projeto DBPedia em extrair e salvar corretamente a informação da Wikipedia em um conjunto de dados estruturados em RDF.

5.3 Análise Geral dos Resultados

Os resultados obtidos através dos experimentos descritos nesse capítulo, bem como através de vários testes informais realizados durante o desenvolvimento do trabalho, que a representação de dados temporais capturados da Web através do modelo de dados interligados RDF, respeitando uma ontologia bem-definida, garante uma altíssima precisão nos resultados encontrados. O fato dos predicados da ontologia conter uma informação semântica embutida é a principal razão da alta precisão, uma vez que basta um predicado ser classificado como temporal para que todas as triplas a ele associadas possam responder perguntas temporais.

Enquanto a precisão é muito alta, a revocação é um pouco menor, principalmente comparando-se os resultados com o que se espera a partir dos dados da Wikipedia. Em parte, isso se deve a simplificações da própria ferramenta desenvolvida, que não teve uma implementação completa e profunda neste momento. Além disso, como a ontologia da DBPedia, que foi utilizada para a construção do conjunto de dados utilizado, não se

especializa em termos temporais, muitas respostas disponíveis na Wikipedia falharam em ser armazenadas no formato de dados interligados.

Contudo, a ferramenta foi capaz de responder a maioria das perguntas feitas e se mostrou de boa qualidade, acima de tudo por ser completa e intuitiva, respondendo perguntas em linguagem natural e retornando todas informações relevantes à pergunta para as quais existem dados armazenados.

6 CONCLUSÃO

As ferramentas de busca atuais têm problemas para explorar o conteúdo temporal dos documentos Web, o que faz com que o resultado das pesquisas com relevância temporal não costuma ter uma qualidade muito boa. Na maioria dos casos, essas ferramentas utilizam apenas uma pequena parte da informação temporal existente nos documentos, especialmente a data de coleta do documento (JIN, 2008).

O projeto DBPedia provê uma ontologia bem-definida para representação de dados da Web em um formato de dados interligados, segundo o modelo RDF. A ontologia define vários predicados, muitos deles especificamente temporais, com semântica embutida. Com isso, os dados podem ser representados de uma maneira simples, flexível e poderosa, por possibilitar consultas de alto nível graças às regras semânticas associadas entre elementos representados. O trabalho de *(GALLINA, 2011a)* contribui com triplas para a DBPedia, obtendo dados temporais de documentos da Web e representando-os utilizando os predicados da ontologia da DBPedia, o que garante uma grande base de dados temporais para serem pesquisados.

A ferramenta desenvolvida possibilitou a funcionalidade de pesquisar esses dados, gerados a partir da ontologia da DBPedia, através de perguntas temporais em linguagem natural em inglês. Há suporte para nove tipos de perguntas, que são automaticamente identificadas e respondidas de acordo com as triplas existentes no conjunto de dados, em linguagem natural. Os experimentos mostraram que a ferramenta retorna resultados de alta qualidade para perguntas referentes a informações temporais que estão representadas no conjunto de dados, seguindo os predicados definidos na ontologia. Além disso, verificou-se que não foi perdido muito em relação aos resultados esperados para os dados originais da Wikipedia, o que mostra que a DBPedia representa grande parte dessas informações e que a ferramenta consegue traduzir a pergunta em uma consulta que de fato encontre esses dados na maioria dos casos.

Há espaço para aperfeiçoamento da ferramenta, de modo a melhorar ainda mais a qualidade os resultados obtidos por ela. Podem ser incluídos mais sinônimos da língua inglesa para suportar mais variações de perguntas, podem ser suportadas referências a tempos presentes e futuros, e o suporte aos predicados pode ser estendido para todos os predicados da ontologia da DBPedia. Além disso, o projeto da DBPedia, o trabalho de (GALLINA, 2011a) e possivelmente outros que apareçam, continuarão evoluindo e gerando cada vez mais triplas com informações temporais, automaticamente melhorando a qualidade dos resultados retornados pela ferramenta desenvolvida neste trabalho, sendo necessário para isso apenas usar os respectivos conjuntos de dados atualizados.

As principais contribuições deste trabalho foram, portanto, as seguintes:

- Identificação de perguntas temporais em linguagem natural em inglês, compatíveis aos predicados da ontologia da DBPedia.
- Conversão de perguntas temporais em consultas SPARQL sobre um conjunto de dados RDF gerado a partir da ontologia da DBPedia.
- Um mecanismo de resposta às perguntas temporais, em linguagem natural em inglês, a partir de um retorno de consulta SPARQL referente aos predicados temporais definidos na ontologia da DBPedia

REFERÊNCIAS

BAEZA-YATES, Ricardo. **Modern information retrieval.** New York : ACM, c1999. 513 p. : il.

BECKETT, D.; BERNERS-LEE, T. (2008). **Turtle - Terse RDF Triple Language.** W3C Team Submission. Disponível em: <http://www.w3.org/TeamSubmission/turtle/>. Acesso em: outubro de 2011.

BERNERS-LEE, T. **Linked data. Design Issues.** Julho, 2006. Disponível em: <http://www.w3.org/De signIssues/LinkedData.html>. Acesso em: novembro de 2011.

SNODGRASS, R. T.; AHN, I. **A Taxonomy of Time in Databases**. In: SIGMOD CONFERENCE. Anais. . . ACM Press, 1985. p.236–246.

GALLINA, L.; GALANTE, R. **Extração e Representação Semântica de Fatos Temporais.** In: Workshop de Teses e Dissertações em Banco de Dados (WTDBD) - XXVI SBBD, 2011a, Florianópolis. Anais do X Workshop de Teses e Dissertações em Banco de Dados (WTDBD) - XXVI SBBD. Porto Alegre : Sociedade Brasieira de Computação, 2011a.

GALLINA, L.; GALANTE, R.; DORNELES, C. F. **Formal Grammar for the Normalization of Relative Temporal Expressions**. In: IADIS International Conference WWW/Internet, 2011b, Rio de Janeiro. Proceedings of the IADIS International Conference WWW/Internet 2011b. Lisboa - Portugal : IADIS - International Association for Development of Information Society, 2011b. p. 373-380.

GRUBER, T. R. **A Translation Approach to Portable Ontologies.** Knowledge Acquisition, v.5, n.2, p.199-200, 1993.

Jena. A semantic Web Framework for Java. Disponível em: <http://jena.sourceforge.net/>. Acesso em: setembro 2011.

JIN, P.; et al. **Tise: A temporal search engine for web contents**. In IITA ’08, pages 220–224, Washington, USA. IEEE Computer Society. 2008.

Joseki. A SPARQL Server for Jena. Disponível em: <http://www.joseki.org/>. Acesso em: setembro 2011.

KLYNE, G.; CARROLL, J. **Resource Description Framework (RDF): Concepts and Abstract Syntax**, World Wide Web Consortium, Recommendation REC-rdf-concepts-20040210, February 2004.

MANICA, E.; GALANTE, R.; DORNELES, C. **Mapeamento Automático de Modelos de Dados XML Temporais Ad-hoc para um Modelo de Dados XML Temporal Padrão**. In ERBD 2010a.

PUSTEJOVSKY, J.; KNIPPEN, R.; LITTMAN, J.; SAURÍ, R. **Temporal and event information in natural language text.** *Language Resources and Evaluation*, 39(2):123–164, May 2005.

TimeML. **Markup Language for Temporal and Event Expressions**. Disponível em: <http://www.timeml.org>. Acesso em: outubro 2011.

TrueKnowledge. The Internet Answering Engine. Disponível em: <http://www.trueknowledge.com/>. Acessado em: outubro 2011.

União Latina - Direção Terminologia e Indústrias da Língua - DTIL. **Línguas e Culturas na Web - Estudo 2007.** Disponível em: <http://dtil.unilat.org/LI/2007/pt/resultados_pt.htm>. Acesso em: dezembro 2011.