Universidade Federal do Rio de Janeiro
Instituto de Matemática
Departamento de Ciência da Computação

Extração de Grafos de Relacionamento Semântico a partir de Estruturas Sintáticas
Uma Abordagem de Melhor Esforço

Danilo Silva de Carvalho

Orientadores:
João Carlos Pereira da Silva (DCC - UFRJ)
André Freitas (DERI - NUIG - Irlanda)

Sumário

Lista de Figuras

Lista de Tabelas

1. Introdução

1.1 Motivação

Estamos vivendo em um momento onde o volume de informação à disposição em meio digital, principalmente na forma textual, cresce em um ritmo muito maior do que estamos aptos a analisar como seres humanos. É portanto cada vez mais necessário o uso do aparato computacional, hoje diretamente atrelado à construção, armazenamento e consumo da informação, para tratar desta informação de maneira a atender nossas necessidades, das mais simples até as mais complexas.

Para isto, é preciso que a informação seja adequada a um conjunto de padrões que possam ser explorados utilizando as técnicas e recursos dos computadores atuais. Esta adequação é chamada de "estruturação da informação" e a informação fora destes padrões é chamada de "não estruturada".

A *Linked Data Web* (Bizer, Heath, Berners-Lee, 2009)[8] – um agrupamento de métodos e conceitos visando a disponibilização da maior quantidade possível de informação estruturada na Web, contextualizada através de suas ligações (links), passou então a ter um papel importante na construção de um novo panorama para a troca de informações na internet. A chamada "Web semântica", traz a ideia de ligar dados para facilitar o acesso por agentes automáticos, que podem consultá-los de forma inteligente e independente.

Os conceitos de *Linked Data* estão hoje presentes em bases de dados abertas na Web, como a DBpedia (Bizer et al., 2009)[9], que são em maioria contruídas a partir de dados estruturados em diferentes formatos, como HTML[1], XML[2] e CSV[3], convertidos para RDF (W3C, 1999)[43]. Estas bases de dados herdam o mesmo tipo de estrutura dos modelos de dados criados manualmente, os quais são representados como ontologias[4] ou vocabulários.

Como a maior parte da informação disponível na Web está em formato não estruturado, a extração de dados estruturados a partir dos não estruturados representa uma oportunidade para um crescimento maior da *Linked Data Web*, beneficiando a todos.

[1] HyperText Markup Language (HTML)

[2] Extensible Markup Language (XML)

[3] *Comma Separated Values*: Formato de representação tabular onde as colunas são separadas por vírgulas.

[4] Mecanismo de especificação para um conjunto de conceitos e os relacionamentos entre eles, ou o modelo de dados que o implementa.

Os trabalhos feitos nesta temática foram focados na extração de ontologias a partir de texto, visando aplicações como sistemas de respostas (question answering systems)[5] ou a construção automática de bancos de dados voltados a domínios específicos. No caso dos sistemas de resposta, a importância central da função de raciocícnio ou inferência, e a dependência de modelos lógicos consistentes associaram a utilidade da extração à necessidades de precisão próximas da perfeição, ou seja, da ausência de qualquer erro no processo de construção destes bancos.

Na construção de bancos de dados para domínios específicos, além da alta precisão, há a necessidade de um nível alto de normalização léxica e estrutural.

Visar os casos onde a precisão elevada é uma necessidade inicial e onde uma pequena inconsistência pode invalidar a utilidade do modelo de dados inteiro desencorajou o uso deste tipo de técnica nos cenários onde uma estratégia de extração semântica por "melhor esforço" (*best-effort*) poderia ser aplicada. Nesta abordagem, o resultado ótimo ou exato não é garantido, ainda que possa ser alcançado, sendo usada em problemas onde o método para a solução exata é muito custoso ou desconhecido. Por apresentar maior tolerância à falhas, este tipo de abordagem vem sendo utilizado com sucesso em sistemas de Recuperação de Informação (IR), onde os resultados são frequentemente avaliados de um ponto de vista qualitativo, impreciso por definição.

A estratégia de "melhor esforço" também poderia ser aplicada à captura do modelo lingístico, facilitando a extração semântica através do "aprendizado" das estruturas sob as quais os processos de extração se apóia direta ou indiretamente.

Recentemente, a IBM demonstrou com seu sistema Watson, que é possível para os computadores responder à perguntas feitas por seres humanos sem nenhum tipo de linguagem especial ou tratamento prévio, vencendo dois campeões humanos em um jogo de perguntas e respostas sobre temas variados (Ferrucci, 2010)[16] (BBC, 2011)[3]. O Watson atraiu grande atenção da imprensa, e muitos usos além daqueles promovidos pelos seus contrutores começaram a ser sugeridos.

Mas apesar do extenso uso de técnicas de processamento de linguagem natural, Watson ainda é, em grande parte, uma demonstração de poder computacional e técnicas de computação de alto desempenho, motivo que levou Noam Chomsky a chamá-lo de "Um rolo compressor maior" (Schmitt, G. C., 2011)[35].

Aparentemente, a opinião de Chomsky remete à limitada, senão ausente, capacidade de interpretação e aprendizado da língua demonstrada por Watson, que se limitava a eliminar respostas dadas como erradas e se adaptar a modelos de resposta, sugerindo uma classificação interna prévia das informações mas não uma adaptação às estruturas da língua exploradas nas perguntas.

O questionamento sobre como trazer a capacidade de interpretação da língua para os computadores levou o autor deste trabalho a buscar soluções no uso de regras envolvendo as estruturas da língua. Esta foi a primeira abordagem adotada por Chomsky em seus trabalhos sobre teoria linguística (Chomsky, 1986)[11].

[5]Sistemas que respondem diretamente à questões feitas pelo usuário, em vez de apresentar uma lista de documentos similares como os Sistemas de Busca.

Um pequeno conjunto destas regras serviria como esqueleto inicial, sobre o qual seriam identificadas "lacunas" de valor semântico, como sujeitos, quantidades, datas, etc. Este conjunto inicial de regras foi implementado como parte do trabalho, sendo um de seus pilares de funcionamento, acompanhado de um analisador eficiente das estruturas da língua, um serviço para consulta à nomes conhecidos, e uma estratégia heurística para substituição de nomes no discurso.

1.2 Descrição do problema

Este trabalho trata do problema de extração de relações semânticas a partir de texto em linguagem natural, ou seja, da forma como é escrito por humanos para humanos.

As relações semânticas expressam fatos envolvendo dois termos do texto. Estes fatos podem ser atributos, ações e estados, entre outros. O conjunto de relacões presentes em um ou mais textos formam uma rede ligando os termos relacionados, esta pode ser representada por um grafo, direcionado ou não, dependendo das necessidades de representação.

Extrair relações envolve a solução de um certo conjunto de problemas relacionados ao processamento de linguagem natural, necessários para alcançar no mínimo as capacidades de identificação das relações e dos termos que estão sendo relacionados. Estes problemas são tratados individualmente no capítulo 3 deste trabalho.

Durante as etapas iniciais do trabalho, foram consideradas duas famílias de línguas para o desenvolvimento: Português e Inglês. O Inglês foi escolhido em função da escassez de recursos de alta qualidade (corpora[6] e parsers[7]) para a língua portuguesa, além de uma maior quantidade de trabalhos existentes para a língua inglesa em relação à portuguesa.

1.3 Objetivo do trabalho

Este trabalho tem como objetivo construir uma ferramenta para extração de relações semânticas na forma de um grafo estruturado, a partir de texto em linguagem natural, usando uma estratégia de "melhor esforço" .

Este grafo estruturado pode ser usado para melhorar a representação de textos por uma perspectiva de Recuperação da Informação e também fornecer uma base de conhecimento comum automaticamente extraída do texto.

O foco em cenários tolerantes a uma extração de melhor esforço, como os de Recuperação de Informação e Sistemas de Respostas em linguagem natural, pode ter um papel fundamental em alcançar um modelo de dados base que pode ser mais tarde usado para permitir um nível de interpretação semântica adequado a cenários mais desafiadores.

[6] Plural de *corpus*: conjunto de textos escritos ou falados de uma língua usados para análise.
[7] Analisadores gramaticais para uma língua.

Do ponto de vista da Recuperação de Informação, o trabalho experimenta uma estrutura de representação dos dados que tem a possibilidade de capturar a semântica do texto, incorporando recursos do RDF (W3C, 1999)[43] como modelo de representação, apesar de não implementá-lo diretamente.

O trabalho concentra-se na Wikipedia como corpus, em suas duas versões de língua inglesa: *English* e *Simple English*. Como consequência, o processo de extração proposto pode também ser usado para enriquecer bases de dados estruturadas já existentes, como a DBpedia (Mendes et al., 2011)[30] e YAGO (Suchanek, Kasneci, Weikum, 2007)[38], que fazem parte da Linked Data Web. Este enriquecimento das bases de dados colabora com o objetivo da Linked Data Web, aumentando a quantidade de informações estruturadas disponíveis para o uso ao redor do mundo. Isto é algo importante, uma vez que a utilidade destas bases de dados é proporcional ao volume e abrangência das informações nelas contidas.

1.4 Organização do trabalho

No segundo capítulo, são apresentados os conceitos linguísticos utilizados ao longo do trabalho e a maneira como se relacionam.

No terceiro capítulo, são abordados os problemas envolvidos na tarefa de extração de relações e como a solução de cada um contribui para o objetivo do trabalho.

No quarto capítulo, são apresentadas as ferramentas utilizadas na solução de cada um dos problemas abordados no terceiro capítulo, e como ocorreu a escolha destas ferramentas.

No quinto capítulo, o modelo de processamento do texto usado no trabalho é descrito em detalhes, especificando as entradas e saídas de cada um dos passos do modelo usado (o pipeline), bem como seus princípios internos de funcionamento.

No sexto capítulo, são exibidas as ferramentas e critérios usados nos experimentos feitos para avaliar o desempenho do trabalho, seguidos dos resultados obtidos na avaliação.

No sétimo capítulo, as conclusões obtidas a partir da avaliação são comentadas, apresentando sugestões para melhorias e expansões ao trabalho.

2. Conceitos: Construção da linguagem

A linguagem natural, linguagem humana natural, ou simplesmente língua, compreende qualquer linguagem desenvolvida pelo ser humano de forma não planejada. O termo é usado em oposição as linguagens artificiais ou construídas, como as linguagens de programação, enquanto que a palavra "humana" é acrescentada para diferenciá-la das demais linguagens usadas por outros animais. A linguagem natural pode ser falada ou escrita.

Segundo Bechara (2004, p. 11)[4], entende-se por língua ou idioma o sistema de simbolos vocais arbitrários com que um grupo social se entende, sendo uma abstração quando considerada fora do ser humano, e quando considerado este, o resultado daquilo que lhe é transmitido pela sociedade.

Sendo uma abstração, podemos nela aplicar transformações da mesma maneira que aplicamos transformações a modelos matemáticos, que são uma forma de abstração de situações da realidade. E para que isto seja possível, é necessária a compreensão do seu processo de construção, que é estudado em profundidade pela gramática, cujo papel é "ordenar os fatos linguísticos da língua padrão na sua época, para servirem às pessoas que começam a aprender o idioma também na sua época" (Bechara, 2004, p. 13)[4].

Neste capítulo, serão apresentados os conceitos linguísticos usados ao longo do trabalho, incluindo as divisões da gramática e seus aspectos, e também técnicas utilizadas para a representação e exploração destes conceitos, como a árvore sintática e o modelo semântico.

2.1 Gramática

A gramática é o conjunto de regras individuais aplicadas a um determinado uso de uma língua e é portando característica de cada língua.

Cabe aqui diferenciar a gramática de linguagem natural da gramática formal, que é um objeto matemático usado para especificar o conjunto de regras de formação de uma linguagem formal, sendo esta uma linguagem construída e portanto sujeita a um conjunto diferente de restrições e premissas. Apesar disto, uma gramática formal pode ser usada para especificar partes de uma linguagem natural, permitindo a aplicação de teorias, modelos matemáticos e ferramentas computacionais das linguagens formais às linguagens naturais.

A gramática de linguagem natural é dividida conforme seus objetos de estudo. Suas principais divisões são apresentadas a seguir, bem como os respectivos papéis destas divisões neste trabalho.

2.1.1 Morfologia

Trata da estrutura, da formação e da classificação das palavras, olhando para elas isoladamente, sem sua participação nas sentenças.

As classificações morfológicas permitem responder questões importantes, como o gênero e classe gramatical (ex: artigo, adjetivo, pronome, etc.), que será mostrado na seção 5.6.2.

2.1.2 Sintaxe

Trata dos padrões estruturais da língua, determinados pelas relações entre as palavras e entre as frases, ou seja, da disposição das palavras na frase e das frases no discurso. Os aspectos da sintaxe relevantes ao trabalho são descritos abaixo:

Estrutura

A formação das sentenças (frases, orações, períodos) se dá por meio da composição de palavras em grupos, chamados sintagmas, que são portanto considerados os constituintes da sentença (Azeredo, 2001, p. 31)[2].

Esta composição ocorre de forma hierárquica, com as palavras sendo o nível mais baixo e compondo os sintagmas, que se combinam formando frases nominais ou orações, dependendo do seu tipo. As orações por sua vez se combinam formando períodos, sendo estes coordenados ou subordinados dependendo de suas relações de dependência (ver abaixo).

Esta estrutura hierárquica permite a representação das sentenças usando árvores, o que possibilita a aplicação de algoritmos de busca conhecidos para percorrer seus constituintes e palavras.

Dependências gramaticais

São relações de dependência existentes entre duas palavras, ou entre duas orações, como a dependência existente entre um adjetivo e o nome por ele modificado (dependência do tipo adjunto adnominal).

As dependências gramaticais estão fortemente ligadas à dependências semânticas, ou seja, dependências entre o significado das palavras. Desta forma, podemos usá-las para resolver ambiguidades e elevar ou diminuir a relevância de candidatos a relações.

Representação

A representação precisa e consistente da linguagem natural é o primeiro passo para torná-la uma abstração "computável", ou seja, explorável utilizando-se de teorias e ferramentas computacionais. Isto é feito através da obtenção de uma gramática formal que contemple uma porção da língua suficiente para cobrir a maior parte as construções utilizadas nos textos que serão analisados, permitindo interpretá-los do ponto de vista sintático.

Esta gramática formal é geralmente uma Gramática Livre de Contexto Probabilistica (PCFG[8]) (Manning, Schütze, 1999)[29] – uma gramática livre de contexto que possui uma probabilidade para cada produção e cuja probabilidade de uma derivação é dada pelo produto das probabilidades das produções usadas na derivação, obtida por meio de aprendizado de máquina supervisionado, sobre um conjunto de textos (corpus) anotado.

Dependendo do tipo de anotação do corpus, a gramática estará associada a um dos dois aspectos sintáticos citados anteriormente: as estruturas de constituintes sintáticos (c-structures na literatura internacional), ou as dependências gramaticais, também conhecidas como funções gramaticais (f-structures), dando origem a uma gramática de constituintes ou uma gramática de dependência respectivamente.

As árvores obtidas pela derivação de uma sentença por um dos tipos de gramática acima são chamadas de árvore de constituintes (ou simplesmente árvore sintática) e árvore de dependências, respectivamente.

Árvore sintática Representa a estrutura de constituintes da sentença.
Neste tipo de árvore, um nó raiz "S" delimita a sentença, os sintagmas são marcados pelos nós não terminais, e as palavras são as folhas, podendo ser marcadas por suas respectivas classificações morfológicas.

As árvores sintaticas são usadas no processo de extração de relações para identificar os componentes que formam uma oração: sujeito, predicado e objeto, bem como complementos nominais.

Exemplo 1:

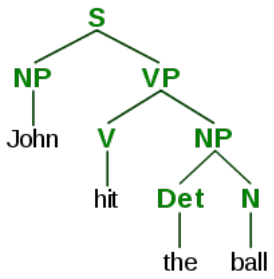


Figura 1: Árv. sintática para a sentença: "'John hit the ball'.
Na figura 1, a sentença é constituida por um sintagma nominal (Noun Phrase - NP) e por um sintagma verbal (Verb Phrase - VP), que por sua vez é constituído de um verbo e um sintagma nominal. As palavras, com exceção de "John", estão marcadas com suas classificações morfológicas: V -> verb (verbo), Det -> Determiner (artigo) e N -> Noun (nome).

[8]Sigla em inglês para Probabilistic Context Free Grammar

Exemplo 2:

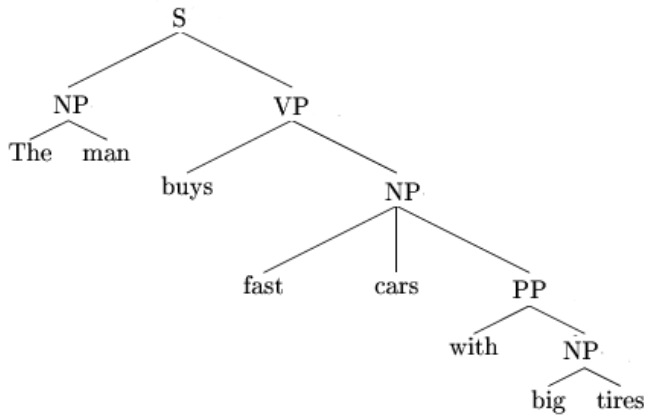


Figura 2: Árv. sintática para a sentença: "The man buys fast cars with big tires".
Na figura 2, a sentença é constituida por três sintagmas nominais (NP), um sintagma verbal (VP) e um sintagma preposicionado (PP). As palavras não receberam marcas morfológicas.

Árvore de dependências Representam a cadeia de dependências gramaticais da sentença.

Na figura 3, as palavras são ligadas por funções de dependência gramatical e a hierarquia é baseada na função representada pelas arestas, onde a palavra modificadora fica um nível abaixo da palavra modificada. As folhas então são palavras não modificadas por nenhuma outra na sentença.

Exemplo 3:

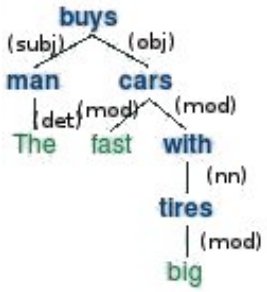


Figura 3: Árv. de dependência para a sentença: "The man buys fast cars with big tires".

Neste exemplo, as palavras "man" e "cars" são dependentes da palavra "buys" pelas funções gramaticais de sujeito e objeto respectivamente. "The" é determinante de "man", "fast" e "with" são modificadores de "cars", "tires" é o nome

referenciado por "with" e "big" é modificador de "tires".

Palavras principais Uma palavra principal é definida informalmente como aquela que melhor representa o significado do constituinte onde ela se encontra.

Em uma árvore de dependência, os nós não-terminais quase sempre são palavras principais, pois os termos abaixo destes são acessórios.

Pode-se usar esta característica para acrescentar marcações de palavra principal a uma árvore sintática, obtendo uma árvore com mais informação sobre a sentença que ela representa. As palavras principais podem ser usadas para evidenciar os termos que participam de uma relação.

Exemplo:

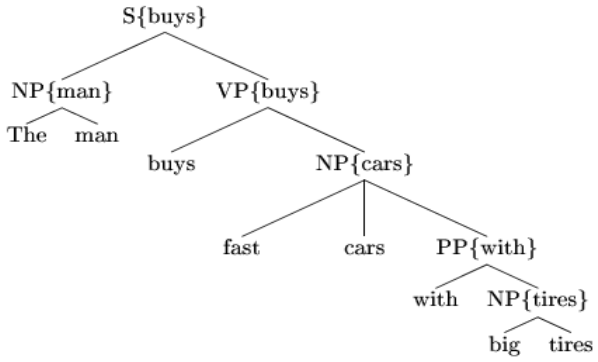


Figura 4: Árv. sintática anotada com palavras principais para a sentença: "The man buys fast cars with big tires".
Na figura 4, palavras principais evidenciam relações entre substantivos em ramos separados da árvore, como "man" e "cars" (relação "buys") e "cars" e "tires" (relação "with").

2.1.3 Semântica

Trata do significado das palavras, tanto isoladamente quanto em frases. É certamente a parte mais dinâmica da gramática, pois é constantemente afetada por elementos que vão além da língua, como fenômenos naturais e sociais.

As palavras muitas vezes apresentam significados diferentes, dependendo do contexto em que são usadas. Por exemplo, em:

I would use this bike if it weren't broken

e

I would buy this bike if I weren't broken,

a palavra *broken* é usada em ambas as frases mas com significado diferente. A análise semântica de uma sentença determina a forma correta de resolver esta abstração.

Dentre os diversos aspectos envolvidos na interpretação correta de uma sentença, tal como a sinonímia, denotação e conotação, o conceito de relação semântica é o predominantemente explorado neste trabalho, pois define um fato

relacionando duas palavras, que pode ser, dentre outros, uma ação, uma característica, ou uma relação de pertinência.

Modelo de relacionamentos

Um modelo de relacionamentos é uma abstração para os relacionamentos semânticos encontrados em textos escritos em linguagem natural, sendo na maioria dos casos uma transformação da abstração da língua para uma abstração lógica ou relacional. Cada modelo de relacionamento distinto possui um grau de complexidade e consegue expressar uma certa quantidade de informações acerca das relações obtidas do texto.

O modelo adotado neste trabalho é o de triplas, onde cada relacionamento é mapeado em uma tupla (sujeito, predicado, objeto), que representa o enunciado lógico **predicado(sujeito, objeto)**. Este é o modelo utilizado entre as aplicações de semântica na web mais populares, que fazem o mapeamento de fontes de informação disponíveis publicamente.

O padrão *Resource Description Framework* (RDF) (W3C, 1999)[43] é um exemplo de representação de relacionamentos usando o modelo de triplas. Ele é usado como modelo de representação da *Linked Data Web* (Bizer, Heath, Berners-Lee, 2009)[8]. A adoção do modelo de triplas permite que este trabalho possa no futuro interoperar com modelos compatíveis com RDF.
O RDF foi construido com o intuito de facilitar a declaração de fatos sobre recursos da Web, como páginas web e outros tipos de documentos. Seu modelo de dados é baseado em conceitos comuns a outras abordagems de modelagem conceitual, como *entidade-relacionamento*, visto que os fatos são representados na forma de expressões (sujeito, predicado, objeto). O modelo de dados do RDF especifica os seguintes conceitos de interesse para este trabalho:

- Classes
  Dividem os recursos em grupos, assim como as classes das linguagems orientadas a objetos, e podem ser descritas usando as *propriedades* RDF. Uma classe é geralmente definida dentro de um certo domínio ou base de dados da Linked Data Web, chamado *namespace*. Desta forma, diferentes bases de dados podem ter classes de mesmo nome sem provocar conflitos. Um exemplo é a classe *dbpedia-owl:Person*, que agrupa todos os recursos referentes à pessoas, declarados na DBpedia.

- Propriedades
  Expressam uma relação entre dois recursos: o sujeito e o objeto. A estrutura que representa a propriedade é a tripla, razão pela qual os termos *propriedade* e *predicado* são intercambiáveis no vocabulário RDF.
  Uma propriedade é sempre direcional: do sujeito para o objeto. A propriedade sempre modifica o sujeito. Como exemplo, há a propriedade *name*, relacionando um recurso (sujeito) a seu nome (objeto).

- Instâncias
  São os recursos membros de uma classe, herdando desta todas as propriedades que descrevem a classe a que pertencem.
  Todos os recursos são instâncias da classe *Resource* e a propriedade *type* desta classe relaciona o recurso (sujeito) à sua classe (objeto).

Como um exemplo, o recurso *Pope_ John_ Paul_ II* declarado na DBpedia, é instância da classe *dbpedia-owl:Cleric*.

- Identificadores universais (URIs) *Uniform Resource Identifiers* (URIs) são cadeias de caracteres (strings) que identificam unicamente um recurso na Web.
  Como exemplo, a URI para o recurso *Pope_ John_ Paul_ II* declarado na DBpedia é *http://dbpedia.org/page/Pope_ John_ Paul_ II*

- Reificação
  Expressam um fato sobre um fato, tratando uma tripla como um recurso. A reificação é feita quando há a necessidade de registrar alguma característica relativa a um fato, como por exemplo quando o fato foi declarado (tempo) ou quem o declarou (autor).

Neste trabalho o modelo de triplas utiliza o conceito de reificação também presente no modelo de representação do RDF. Isto permite o registro de informações como a data em que ocorreu determinado fato.

Grafo de relações

O grafo de relações é uma representação para o modelo de relacionamentos, onde sujeitos e objetos são os nós do grafo e os predicados são as arestas. A análise do grafo de relações permite descobrir fatos que estão implicitos no texto, percorrendo caminhos entre nós. Trabalhos em desenvolvimento buscam usar as informações contidas nestes grafos para responder a consultas em linguagem natural (Freitas et al., 2011)[21].

As reificações são representadas neste trabalho como arestas especiais, e são desenhadas como linhas pontilhadas. Abaixo, um exemplo de grafo de relações para uma sentença:

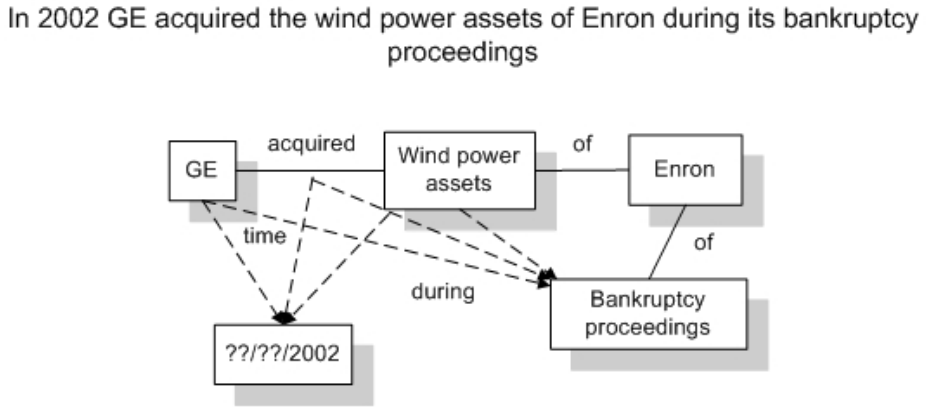


Figura 5: Grafo de relações para uma sentença

Entidades Nomeadas (NEs)

Entidades nomeadas (Named Entities - NEs) são coisas ou pessoas que podem ser referenciadas por termos, simples ou compostos, de conhecimento público. Por exemplo, o termo "Cristo Redentor" refere-se a uma determinada estátua e "Napoleão" refere-se a uma determinada pessoa. Estes termos correspondem a entidades (físicas ou abstratas) do mundo real.

Uma NE pode ser referenciada por mais de um termo, como em "Bill Gates" e "William Henry Gates III", e a correta atribuição de um termo à sua entidade correspondente é essencial para a definir com precisão a quem ou a que se refere um determinado fato. Esta tarefa de atribuição é chamada de Reconhecimento de Entidades Nomeadas (Named Entity Recognition - NER).

Ontologias

No contexto de Ciência da Computação, uma ontologia é um mecanismo de especificação, que define uma forma de representar um modelo de domínio ou discurso (Liu, Özsu, 2009)[28].

As ontologias desempenham um papel vital na *Web semântica*, onde são usadas para representar toda a informação disponível nas bases de dados, utilizando a linguagem OWL . Esta linguagem, juntamente com o RDF e outras tecnologias, formam a arquitetura conhecida como *Semantic web stack*, ilustrada na figura abaixo:

As ontologias utilizadas neste trabalho representam as informações sobre um conjunto de entidades nomeadas na forma de um grafo de relações, onde as NEs são ligadas por meio de predicados, que representam fatos sobre elas.

Um exemplo de ontologia é a DBPedia, que reúne dados sobre as entidades registradas na Wikipedia, a mais popular enciclopédia da internet (Bizer et al., 2009)[8].

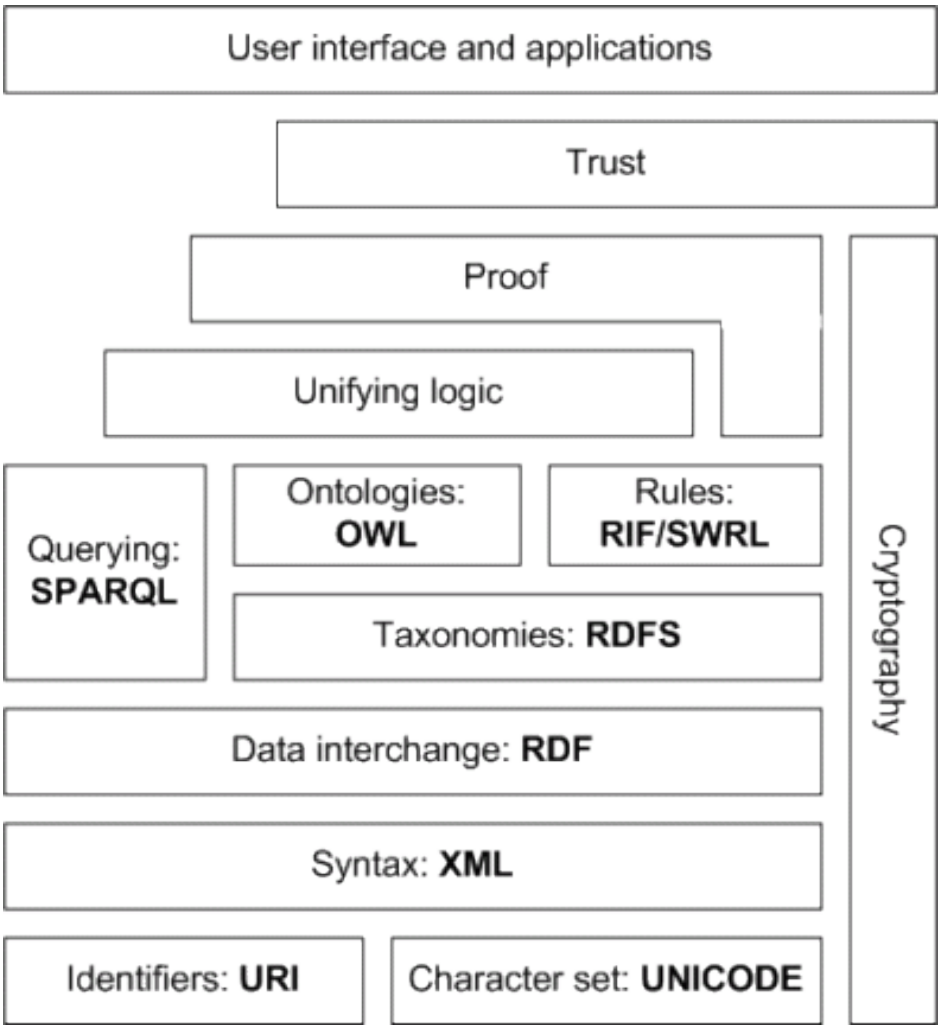


Figura 6: Semantic web stack

2.1.4 Fonética

Trata do sistema sonoro da língua, classificando e determinando o papel dos diferentes sons na articulação das palavras e seus significados.

Para os sistemas envolvidos na síntese de voz a partir de texto, foram desenvolvidos vários conjuntos de marcações, com o objetivo de conseguir uma comunicação mais agradável e confiável com o usuário (W3C, 2004)[47]. Estas marcações podem ser também usadas para auxiliar na interpretação de texto, principalmente na eliminação de ambiguidades que não podem ser resolvidas por uma gramática formal.

Como este trabalho foi voltado ao processamento de textos da internet, já disponibilizados em forma escrita, não foi feito nenhum tratamento no texto do ponto de vista fonético.

3. Problemas a tratar

O problema central – extrair relações semânticas a partir de texto em linguagem natural, sugere algo difícil de ser tratado quando enunciado desta forma. Mas ele pode ser dividido em uma série de problemas menores, identificados na elaboração deste trabalho, cujas soluções visam responder às seguintes perguntas, que são os maiores obstáculos no caminho do objetivo:

1. O que é ou pode ser considerado uma relação?
2. Onde uma relação pode ser encontrada no texto?
3. Quais são os termos que podem estar em uma relação?
4. Onde no texto estão estes termos?
5. De que forma(s) estes termos são ligados pela relação?
6. Como representar uma relação no computador?

Estas perguntas podem ser traduzidas para problemas conhecidos do processamento de linguagem natural. São eles:

3.1 Resolução de Entidades Nomeadas (NER)

É a tarefa de atribuir uma identidade a um termo (nome ou expressão) quando este se refere a uma Entidade Nomeada (NE). Saber que "Barack Obama" e "The current AMD CEO" se referem à dois indivíduos é importante para decidir sobre onde encontrar estes indivíduos no texto, dado que eles podem estar presentes em alguma relação. Com isto as perguntas 3 e 4 podem ser respondidas.

3.2 Resolução de correferência

É o mapeamento dos termos anteriormente citados no texto ou fora dele, que são referenciados geralmente na forma de pronomes. Por exemplo, em "The show ended at six and we came back by bus. It was really good.", o pronome "It" na segunda frase refere-se ao termo "The show" citado na primeira frase.

Assim como a NER, este mapeamento é importante para encontrar no texto os termos que podem estar em relações, respondendo a pergunta 4.

3.3 Normalização de referências

É a tarefa de garantir que todas as referências a uma mesma entidade estão representadas da mesma forma. Com isso busca-se garantir a resposta às perguntas 4 e 6, fazendo com que os termos sejam sempre encontrados e que possam ser representados consistentemente.

3.4 Análise sintática (*parsing*)

É a transformação da língua em sua representação estrutural hierárquica na forma de árvore para uma gramática específica. Nela estarão discriminados os constituintes das sentenças e todas as suas palavras com suas respectivas classificações gramaticais.

Resolver este problema garante a resposta às perguntas 4 e 5, ao identificar e classificar os constituintes e suas ligações, e permite a resposta às perguntas 1, 2 e 3 ao explicitar a estrutura sobre a qual as relações são expressas na língua.

3.5 Extração de relações

É a obtenção da representação das relações encontradas no texto em um *modelo de relacionamentos*, permitindo seu uso por qualquer aplicação que possa fazer uso do modelo. Corresponde à resposta de todas as perguntas, mas principalmente as perguntas 1, 2 e 6.

Após localizar e classificar um termo ou expressão como uma relação, as demais perguntas servirão para preencher as lacunas do modelo de relacionamento (a representação), incluindo os fatos sobre a relação (reificação). Uma vez preenchido, o modelo precisa apenas ser transformado para uma apresentação adequada aos fins do usuário.

4. Trabalhos utilizados

O trabalho foi iniciado com uma pesquisa acerca dos estudos e ferramentas mais recentes na área de processamento de linguagem natural, com o objetivo de identificar os avanços já conseguidos na solução de cada um dos problemas a tratar. Esta pesquisa levou à seleção de um conjunto de trabalhos como candidatos à *melhor*[9] solução destes problemas. São apresentados abaixo os trabalhos pesquisados sobre os problemas citados, a avaliação feita e a consequente escolha daqueles utilizados neste trabalho.

4.1 DBpedia

Projeto comunitário iniciado por (Bizer et al., 2007)[7] para extrair conteúdo estruturado da informação disponibilizada na Wikipedia. Esta informação estruturada é então disponibilizada na Web, na forma de uma ontologia, sobre a qual os usuários podem fazer consultas e obter os dados em formatos abertos, como o RDF.

4.2 Spotlight

Ferramenta para anotar menções à entidades presentes na DBpedia que sejam encontradas em textos (Mendes et al., 2011)[30], efetivamente realizando a tarefa de *NER*. Possui um serviço público na web que responde à requisições com texto em linguagem natural e parâmetros de controle, produzindo como saída documentos RDFa (W3C, 2004)[46], XHTML (W3C, 2000)[44] ou JSON (Crockford, 2006)[13] com as entidades identificadas marcadas com suas respectivas URIs na Dbpedia. Este serviço é constantemente alimentado com os dados disponibilizados na DBpedia, usando técnicas de aprendizado de máquina para realizar suas funções.

4.3 NLTK

O Natural Language Toolkit (NLTK) (Bird, Klein, Loper, 2009)[6] é uma biblioteca de programação para a linguagem *Python* contendo pacotes de dados linguisticos (corpora, gramáticas e exemplos) e código para tratamento de linguagem natural, incluindo parsers e visualizadores.

4.4 ReVerb

Extrator de relações que utiliza uma combinação de CRF (Lafferty, McCallum, Pereira, 2001)[27] e expressões regulares, aplicadas tanto nas frases diretamente (relation phrases) como nas *POS tags*[10] (POS patterns) (Fader, Soderland, Etzioni, 2011)[15]. O pacote do ReVerb vem com o executável e a biblioteca,

[9] Segundo os critérios de avaliação utilizados neste trabalho (seção 4.8).

permitindo usá-lo sozinho, passando arquivos de texto como entrada, ou dentro de um programa em linguagem Java , através de sua API[11].

O ReVerb também utiliza o modelo de triplas para representar relacionamentos. O executável recebe como entrada texto em linguagem natural e gera como saída arquivos de texto com campos separados por tabulações. Os campos identificam as triplas e o grau de confiança das relações, entre outros dados. As linhas separam as sentenças encontradas.

Devido às técnicas utilizadas, o ReVerb possui uma tendência a identificar relações longas (com muitas palavras), envolvendo expressões, como em "Neil Armstrong was the first man to step on the moon." -> sujeito: "Neil Armstrong", objeto: "the moon", predicado: "was the first man to step on".

4.5 LBJ

Pacote de resolução de correferência da Universidade de Illinois (Bengtson, Roth, 2008)[5]. Reconhece pessoas, organizações e entidades geopolíticas. Além das identidades, trata de relações semânticas tais como sinônimos e antônimos. Pode ser treinado com um corpus anotado no formato ACE[12], mas o executável já vem treinado.

Assim como o ReVerb, recebe como entrada arquivos contendo texto em linguagem natural e produz como saída arquivos de texto com as entidades marcadas com identificadores (números inteiros) e suas referências.

4.6 Reconcile

Pacote de resolução de correferência da Universidade de Utah (Stoyanov et al., 2010)[37]. Pode ser treinado com um corpus anotado no formato ACE (Peters, Smith, 1986)[34] ou MUC[13] (NIST, 1987)[33] SGML[14] (ISO 8879:1986)[1], mas o executável já vem treinado.

Também recebe como entrada texto em LN. Produz como saída documentos XML (W3C, 1996)[42] com as entidades identificadas através de números (NO) e com as identidades a que referenciam (CorefID).

4.7 Stanford CoreNLP

Pacote de processamento de linguagem natural da Universidade de Stanford (Klein, Manning, 2003)[26]. Contém gramáticas e parsers para uma variedade de línguas.

O principal componente do CoreNLP usado neste trabalho é o Stanford Parser, um analisador sintático baseado em PCFG, com foco na língua inglesa, e otimizações diversas.

[10] Part of Speech (POS) tag é uma marcação indicando a classe gramatical da palavra por ela marcada.

[11] Application Programming Interface (API) é uma especificação de programação estabelecida por um software para permitir a utilização de suas funcionalidades, sem que o acesso a seu código fonte seja necessário.

[12] Australian Corpus of English (ACE)

[13] Message Understanding Conference (MUC)

[14] Standard Generalized Markup Language (SGML)

4.8 Avaliação: desempenho relativo aos objetivos e premissas

As ferramentas foram avaliadas em grupos, conforme suas funções, para escolher aquelas que melhor realizavam as tarefas para as quais foram projetadas. O desempenho foi medido através do número de erros e acertos, avaliados por um revisor humano, em cada uma das instâncias em que as funções das ferramentas eram requisitadas (ex: em cada pronome no caso do LBJ e Reconcile).

A impressão obtida pelo avaliador após a primeira dezena de instâncias colhidas aleatoriamente em cada um dos dois textos avaliados (artigos da Simple English Wikipedia sobre Barack Obama e New York) foi usada como critério de decisão. Outro critério levado em conta foi o desempenho da extração em termos de velocidade e consumo de memória.

A ideia foi a de obter a melhor ferramenta disponível para a solução de cada um dos problemas envolvidos na extração de relações. Sendo feitas as seguintes comparações:

- Análise Sintática: NLTK x Stanford parser
- Resolução de correferência: Reconcile x LBJ

Até o término do desenvolvimento deste trabalho, nenhuma ferramenta publicamente disponível para NER além do Spotlight foi encontrada. Desta forma o Spotlight não pode ser comparado com outras ferramentas.

O mesmo ocorreu com o ReVerb, sendo a única ferramenta disponível ao público para tratar do problema de extração de relações isoladamente, ou seja, sem nenhum tratamento prévio, e que oferecia a saída desejada para o trabalho (triplas).

Não foi encontrada uma ferramenta pública para a tarefa de normalização de referências.

4.8.1 Ferramentas escolhidas

Após as comparações, as seguintes ferramentas foram escolhidas:

- Análise sintática -> Stanford parser
- Resolução de correferência -> Reconcile
- Resolução de Entidades Nomeadas -> Spotlight
- Extração de relações -> ReVerb

O Stanford parser foi escolhido para a tarefa de análise sintática porque apresentou tanto um melhor número de acertos quanto um melhor desempenho computacional. Isto foi devido ao conjunto de otimizações feitas no Stanford parser, tanto no código quanto na gramática, e ao fato do NLTK não incluir uma gramática completa para a língua inglesa.

Para a tarefa de resolução de correferência foi escolhido o Reconcile, que obteve um numero de acertos maior que o LBJ e um desempenho computacional muito superior.

Sem competidores, o Spotlight e o ReVerb foram escolhidos para as tarefas de NER e extração de relações, respectivamente.

5. Modelo de processamento do texto: o pipeline

5.1 Motivação

A avaliação das ferramentas escolhidas mostrou que nenhuma delas foi planejada para operar em conjunto com outras ferramentas, ou seja, não havia uma forma de controlar o formato das saídas para tentar compatibilizá-las. Isto trouxe dúvidas quanto ao modelo de processamento de texto que seria utilizado para compor as anotações e chegar ao resultado desejado.

A primeira ideia cogitada foi criar um formato comum de anotações, baseado em XML, para o qual seriam transformadas todas as saídas e em seguida reunidas em um único arquivo. Mas isto não se mostrou viável, dadas as grandes diferenças e incompatibilidades entre os formatos de cada ferramenta.

Como todas as ferramentas tinham em comum o mesmo formato de entrada (texto livre), a solução adotada então foi a de uma linha de montagem (pipeline), onde a saída de uma ferramenta é transformada para o formato de entrada da próxima na linha.

A ideia geral do pipeline é a de que ainda que os problemas tratados sejam independentes, a solução de cada problema pode ser facilitada caso sejam resolvidos em uma certa ordem e as etapas de solução possam ser encadeadas de forma a aproveitar os resultados das etapas anteriores. Esta característica foi observada nos problemas que compõem a extração de relações, e a melhor ordem encontrada para o conjunto de ferramentas selecionadas foi a seguinte:

1. Resolução de Entidades Nomeadas
2. Resolução de correferência
3. Análise sintática
4. Extração de relações

O modelo pipeline traz alguns benefícios:

- Facilidade de substituir ou acrescentar elementos na linha.
- Possibilidade de aplicar melhorias nas transformações, independente de formato.
- Facilidade na identificação de gargalos dependentes da entrada.
- Melhorias em um elemento da linha são passadas para os próximos.

Esta última vantagem tornou-se um fator decisivo para a escolha deste modelo.

A maior desvantagem deste modelo, a dificuldade em identificar a origem de um erro após o processamento completo, é amenizada pelo fato da maior parte dos erros encontrados serem facilmente caracterizáveis, uma vez que as etapas do pipeline tratam de problemas bem distintos.

5.2 Ligando as ferramentas

Antes da montagem inicial do pipeline, algumas mudanças foram feitas para corrigir problemas observados ainda na etapa de avaliação das ferramentas, de forma a evitar a propagação de erros.

Para a tarefa de NER, foi desenvolvido o programa *envia_spotlight*, que facilita a parametrização das chamadas ao serviço do Spotlight, o que possibilitou maximizar a precisão desta tarefa para a classe de entradas utilizadas no trabalho.

Para a tarefa de resolução de correferência, inicialmente foi escolhido o Reconcile. Mas uma análise mais cuidadosa revelou alguns problemas recorrentes nos resultados do Reconcile, causados por características do seu algoritmo de resolução. Entre estes problemas estavam a ausência de distinção de gênero e um limite de distância máxima (número de palavras) entre um termo e sua correferência.

Estas limitações levaram ao desenvolvimento de um programa para resolução de correferências pronominais como parte do trabalho, que foi acoplado ao programa normalizador, dando origem ao *norm_ne_coref* (seção 5.4).

Após os testes com o conjunto inicial Spotlight -> norm_ne_coref -> Stanford parser -> ReVerb, foi decidido que o ReVerb não possuia a flexibilidade desejada para o trabalho, optando-se pela construção de um extrator como parte do trabalho. Este extrator foi implementado na forma do programa *extrai_rels_cstruct* (seção 5.6).

O funcionamento do pipeline pode ser resumido no seguinte diagrama:

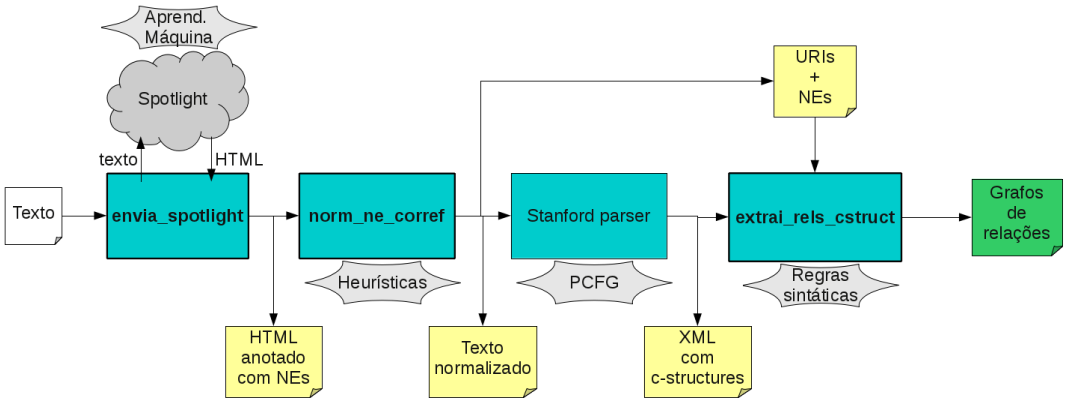


Figura 7: Diagrama de funcionamento do pipeline: elementos do pipeline: *envia_spotlight*, *norm_ne_coref*, *Stanford parser* e *extrai_rels_cstruct*;
saídas: HTML anotado com NEs, texto normalizado, XML com c-structures e arquivo com URIs + NEs;
resultado: Grafos de relações.
Junto aos elementos estão seus princípios de funcionamento: Aprend. máquina (Spotlight), heurísticas (norm_ne_coref), PCFG (Stanford parser) e regras sintáticas (extrai_rels_cstruct).
Os elementos *envia_spotlight*, *norm_ne_coref*, *extrai_rels_cstruct* (em negrito) foram desenvolvidos no trabalho.

A ligação entre as ferramentas foi feita pelo aproveitamento das anotações produzidas por elas, que são usadas para criar informações adicionais no próximo passo do pipeline. Estas informações podem ser a identificação de uma entidade nomeada, o gênero de um nome, ou a árvore sintática para uma sentença. As informações são acrescentadas ao texto original quando necessário, por exemplo substituindo referências diferentes a uma mesma pessoa por uma referência única, como no caso "William Henry Gates III" -> "Bill Gates".

5.3 Spotlight

5.3.1 Uso e Parâmetros

O Spotlight (Mendes et al., 2011)[30] é utilizado através do programa *envia_spotlight*, construído como parte do trabalho, sendo o primeiro passo do pipeline. Este programa recebe como único argumento o local de um arquivo de texto, que será enviado ao Spotlight. O programa envia o conteúdo do arquivo especificado para o serviço web do Spotlight via HTTP POST, com os seguintes parâmetros:

- "confidence" -> "0.2" (grau de confiança ou precisão)
- "support" -> "100" (relevância aplicada ao contexto)
- "spotter" -> "CoOccurrenceBasedSelector" (tipo de classificador)
- "disambiguator" -> "Document" (tipo do desambiguador)

O classificador baseado em co-ocorrências reforça a classificação de entidades usando termos que aparecem próximos (co-ocorrem) com frequência, demostrando nos testes ser o mais eficiente, enquanto o desambiguador por documento é a única opção disponível para textos longos. A confiança e o suporte foram sendo ajustados ao longo dos testes até chegar a valores ótimos, que não comprometiam muito a *precisão*[15] nem o *recall*[16] das entidades encontradas.

O Spotlight então responde com um documento HTML onde as entidades encontradas estão marcadas com links para suas respectivas URIs na DBpedia. O programa escreve o conteúdo deste documento na saída padrão (stdout), que é direcionada para um arquivo no momento de sua execução.

A saída deste passo do pipeline é um documento HTML com as NEs encontradas no texto original anotadas em âncoras (<a>).

Exemplo de entrada e saída para este passo do pipeline:

Entrada:

Barack Hussein Obama II (born August 4 1961) is the 44th and current president of the United States of America.
Obama was born in Honolulu, Hawaii. His father was a black foreign student from Kenya and his mother was a white woman from Kansas.

[15] Fração das entidades encontradas com correspondência correta na ontologia (DBpedia).
[16] Fração das entidades presentes no texto que são encontradas.

Saída:

```
<a about="http://dbpedia.org/resource/Barack_Obama"
   href="http://dbpedia.org/resource/Barack_Obama"
   title="http://dbpedia.org/resource/Barack_Obama"
   target="_blank" >

   Barack Hussein Obama II
</a>
(born
<a about="http://dbpedia.org/resource/August_4"
   href="http://dbpedia.org/resource/August_4"
   title="http://dbpedia.org/resource/August_4"
   target="_blank" >

   August 4
</a>
1961) is the 44th and current president of the United States of
<a about="http://dbpedia.org/resource/Americas"
   href="http://dbpedia.org/resource/Americas"
   title="http://dbpedia.org/resource/Americas"
   target="_blank" >

   America
</a>
.<br/><br/>
<a about="http://dbpedia.org/resource/Barack_Obama"
   href="http://dbpedia.org/resource/Barack_Obama"
   title="http://dbpedia.org/resource/Barack_Obama"
   target="_blank" >
Obama
</a>
was born in
<a about="http://dbpedia.org/resource/Honolulu"
   href="http://dbpedia.org/resource/Honolulu"
   title="http://dbpedia.org/resource/Honolulu"
   target="_blank" >

   Honolulu, Hawaii
</a>
. His father was a black

<a about="http://dbpedia.org/resource/International_student"
   href="http://dbpedia.org/resource/International_student"
   title="http://dbpedia.org/resource/International_student"
   target="_blank" >

   foreign student
</a>
from
```

```
<a about="http://dbpedia.org/resource/Kenya"
   href="http://dbpedia.org/resource/Kenya"
   title="http://dbpedia.org/resource/Kenya"
   target="_blank" >

   Kenya
</a>
and his mother was a white woman from
<a about="http://dbpedia.org/resource/University_of_Kansas"
   href="http://dbpedia.org/resource/University_of_Kansas"
   title="http://dbpedia.org/resource/University_of_Kansas"
   target="_blank">

   Kansas
</a>.
```

No exemplo acima, foram encontradas sete entidades nomeadas: *Barack Hussein Obama II*, *August 4*, *America*, *Honolulu, Hawaii*, *foreign student*, *Kenya* e *Kansas*. Elas foram anotadas com suas respectivas URIs (atributo *about*). Como pode ser observado, há um erro na URI atribuída à entidade *Kansas*. Isto será tratado na seção *6.2.1*.

5.4 Normalizador de referências

O normalizador foi originalmente construído para transformar a saída do Spotlight em texto puro, substituindo as diversas ocorrências diferentes de uma determinada entidade por uma única representação. Mas ao longo do andamento do trabalho, acumulou também a função de resolver correferências, em função de deficiências encontradas na ferramenta escolhida para este papel (seção 4.2).

A implementação do normalizador foi feita no programa *norm_ne_coref* e é baseada em uma simples estratégia heurística, envolvendo o posicionamento dos nomes e referências ao longo do texto.

5.4.1 Funcionamento

O normalizador recebe como argumentos o local de um arquivo em linguagem de marcação, o nome da tag (anotação) que marca as entidades, o nome do atributo que identifica as entidades em suas tags e o nome do atributo que identifica correferências, se este existir no documento.
No caso da saída do Spotlight estes argumentos são respectivamente:

1. O nome do arquivo para onde foi direcionada a saída do *envia_spotlight*.
2. "a" (âncora HTML).
3. "about" (atributo especial acrescentado pelo Spotlight, com a URI da entidade).
4. "about" (como contém um identificador único, também refere-se às outras instâncias da mesma entidade no texto).

Exemplo

Supondo que a saída apresentada no exemplo do último passo (Spotlight) foi direcionada para um arquivo com nome *obama_ner.html*, a chamada ao *norm_ne_coref* ficaria da seguinte forma:

norm_ne_coref obama_ner.html a about about •

Além disto, o normalizador faz uso de uma lista de nomes obtida de um censo estadunidense (Census, 2010)[10], onde constam praticamente todos os nomes de cidadãos dos EUA, separados por gênero, e de uma lista de pronomes da língua inglesa, também separados por gênero. As listas são usadas para a identificação do gênero dos nomes e pronomes encontrados nos textos.

Ele processa o documento especificado em duas etapas:

1. Normalizar as entidades
   Nesta primeira etapa, ele procura pelas tags marcando as entidades e armazena a primeira referência encontrada para cada entidade, associada ao seu respectivo identificador (URI). As demais referências encontradas ao longo do texto apontando para o mesmo URI são substituidas pela referência armazenada. Isto uniformiza a representação de todas as entidades no texto, tornando a próxima etapa possível.

   No exemplo de saída do Spotlight, a primeira entidade nomeada encontrada, *Barack Hussein Obama II*, é associada a URI *http://dbpedia.org/resource/Barack_Obama*. A partir deste ponto, todos os nomes no texto que estejam associados a esta mesma URI serão substituidos pela primeira referência: *Barack Hussein Obama II*

2. Resolver as correferências
   Nesta etapa o texto é percorrido palavra por palavra. Caso seja encontrada uma marcação de entidade, o nome desta é buscado na lista de nomes para saber seu gênero e em seguida é colocado em uma pilha junto com a informação de gênero (M, F ou desconhecido).

   Caso seja encontrado um pronome, este é substituido pelo nome mais alto na pilha que possua o mesmo gênero, sem alterar a pilha.

   Caso três pronomes tenham sido encontrados sem que o nome no topo da pilha tenha sido referenciado, o nome no topo da pilha é descartado. Isto ocorre até que reste apenas um nome na pilha, que não é descartado.

   Prosseguindo com o exemplo anterior, ao encontrar o nome *Barack Hussein Obama II*, este é acrescentado a pilha com o seu gênero obtido da lista de nomes. O mesmo ocorre com *America* e *Honolulu, Hawaii*. Ao encontrar o pronome *His*, a pilha está no estado abaixo:

<Honolulu, Hawaii | desconhecido>
<America | desconhecido>
<Barack Hussein Obama II | M[17]>

> O pronome *His* será então substituido pelo nome de Obama, o mais alto na pilha do gênero masculino. Os demais pronomes são tratados analogamente.

Na etapa de resolução das correferências, apenas os pronomes pessoais retos e oblíquos (He, They, Me) e os pronomes possessivos (His, Yours) são considerados. Estes pronomes cobrem a maior parte dos casos envolvendo entidades nomeadas.

A lógica da segunda etapa supõe uma hierarquia simples de nomes no discurso, ou seja, o uso de pronomes para se referir ao nome mais próximo citado anteriormente, seguido do segundo mais próximo e assim por diante. Este é o caso da grande maioria dos textos, motivo pelo qual o normalizador obteve um melhor desempenho em relação ao Reconcile (Stoyanov et al., 2010)[37] e LBJ (Bengtson, Roth, 2008)[5] na maioria dos testes, onde os últimos ignoravam pronomes com maior frequência.

Após a normalização e resolução de correferências, o texto com as entidades e pronomes substituídos é escrito na saída padrão, que novamente é direcionada para um arquivo no momento da execução.

A saída deste passo é o texto com pronomes substituídos pelos nomes a que se referem, com uma representação única para cada entidade nomeada, e um arquivo com as URIs encontradas e suas respectivas entidades.

Exemplo

Ainda supondo o uso do exemplo de saída do Spotlight como entrada do normalizador, a saída obtida seria:

Barack Hussein Obama II (born August 4 1961) is the 44th and current president of the United States of America.
Barack Hussein Obama II was born in Honolulu, Hawaii. Barack Hussein Obama II father was a black foreign student from Kenya and Barack Hussein Obama II mother was a white woman from Kansas.

As substituições feitas estão sublinhadas. •

5.4.2 Aproveitamento dos passos anteriores

O *norm_ne_coref* aproveita as marcações do Spotlight obtidas pelo *envia_spotlight* para tornar uniformes as representações das entidades encontradas, simplificando o texto e tornando possível a tarefa de resolver os pronomes de forma consistente, usando a informação de gênero.

É comum haver diferentes nomes para uma entidade que dificultam a caracterização de seu genero, como "Barack Obama" e "president". Além disso, por uma questão de simplificação, apenas os nomes identificados como entidades são tratados em vez de todos os nomes, o que evita o tratamento de casos complicados como "August" (nome ou mês?) e "Nature" (substantivo comum ou marca?).

[17] Para efeito de definição do gênero, são considerados apenas o primeiro e segundo nomes de uma entidade, nesta ordem de prioridade.

5.5 Stanford parser

5.5.1 Uso e Parâmetros

O Stanford parser (Klein, Manning, 2003)[26] é utilizado diretamente, rodando seu executável sobre um ou mais arquivos de texto produzidos pelo *norm_ne_coref*. Os parâmetros utilizados são:

1. "annotators" -> "tokenize,ssplit,pos,lemma,ner,parse" (tipos de anotações que serão feitas)[18].
2. "file" -> nome do arquivo de texto produzido pelo *norm_ne_coref*.
3. "filelist" -> arquivo contendo uma lista de nomes de arquivos (no caso de mais de um).
4. "outputDirectory" -> diretório onde ficarão os arquivos de saída do parser.

O parser processa os arquivos de texto especificados, separando as sentenças e obtendo suas árvores sintáticas (c-structures) que são gravadas em strings no formato *Penn Treebank* (CIS/UPenn, 1995)[41] dentro de arquivos XML contendo todas as anotações. O parser produz um arquivo XML por texto anotado.

A saída deste passo é o documento XML contendo, a representação das árvores sintáticas de todas as sentenças do texto produzido pelo passo anterior, no formato *Penn Treebank*.

Exemplos

Supondo o aproveitamento do exemplo de saída do *norm_ne_coref*, a saída para o Stanford parser seria:

Primeira sentença:

Barack Hussein Obama II (born August 4 1961) is the 44th and current president of the United States of America.

Saída:

```
<parse>
    (ROOT
        (S
            (NP (NP (NNP Barack) (NNP Hussein) (NNP Obama) (NNP II))
            (PRN (-LRB- -LRB-)
                (VP (VBN born)
                    (NP-TMP (NNP August) (CD 4) (, ,) (CD 1961))
                )
                (-RRB- -RRB-)
            )
        )
        (VP (VBZ is)
            (NP (NP (DT the) (ADJP (JJ 44th) (CC and) (JJ current)) (NN president))
                (PP (IN of)
```

[18] A única anotação de interesse para este trabalho é o "parse", mas as outras são pré-requisitos e precisam ser especificadas.

```
                    (NP (DT the) (NNP United) (NNPS States))
                )
                (PP (IN of)
                    (NP (NNP America))
                )
            )
        )
        (. .)
    )
    )
</parse>
```

Árvore sintática equivalente:

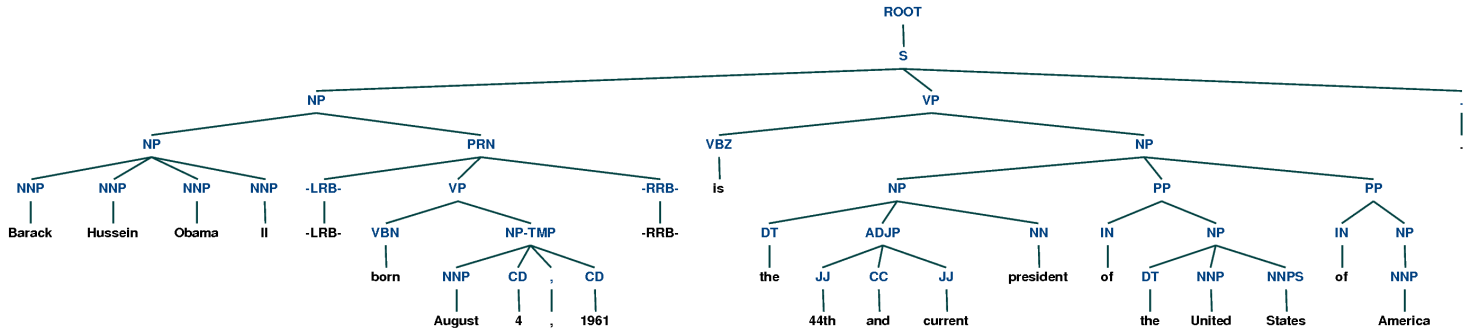


Figura 8: Exemplo de saída do Stanford parser para a sentença: *Barack Hussein Obama II (born August 4 1961) is the 44th and current president of the United States of America.*

Segunda sentença:

Barack Hussein Obama II was born in Honolulu, Hawaii.

Saída:

```
<parse>
    (ROOT
        (S
            (NP (NNP Barack) (NNP Hussein) (NNP Obama) (NNP II))
            (VP (VBD was)
                (VP (VBN born)
                    (PP (IN in)
                        (NP (NNP Honolulu) (, ,) (NNP Hawaii))
                    )
                )
            )
            (. .)
        )
    )
```

```
</parse>
```

Árvore sintática equivalente:

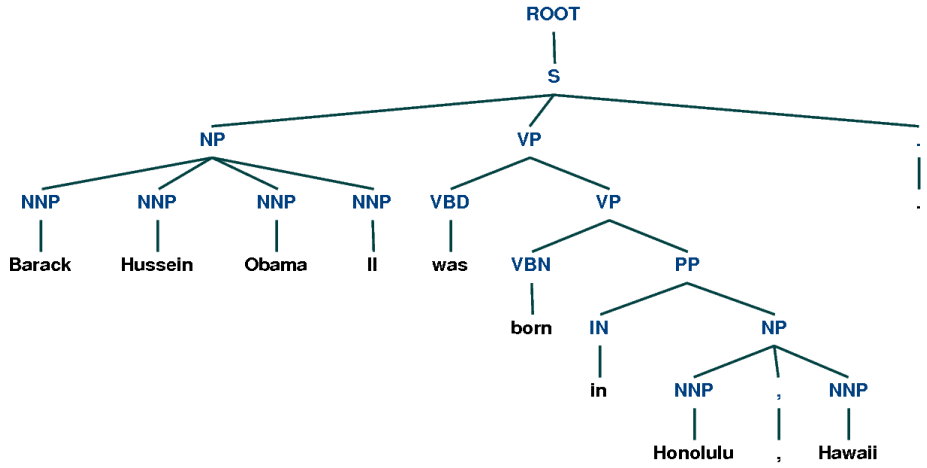


Figura 9: Exemplo de saída do Stanford parser para a sentença: *Barack Hussein Obama II was born in Honolulu, Hawaii.*

Terceira sentença:

Barack Hussein Obama II father was a black foreign student from Kenya and Barack Hussein Obama II mother was a white woman from Kansas.

Saída:

```
<parse>
    (ROOT
        (S
            (S
                (NP (NNP Barack) (NNP Hussein) (NNP Obama) (NNP II) (NN father))
                (VP (VBD was)
                    (NP (NP (DT a) (JJ black) (JJ foreign) (NN student))
                        (PP (IN from)
                            (NP (NNP Kenya))
                        )
                    )
                )
            )
            (CC and)
            (S
                (NP (NNP Barack) (NNP Hussein) (NNP Obama) (NNP II) (NN mother))
                (VP (VBD was)
                    (NP (NP (DT a) (JJ white) (NN woman))
```

```
                        (PP (IN from)
                            (NP (NNP Kansas))
                        )
                    )
                )
            )
            (. .)
        )
    )
</parse>
```

Árvore sintática equivalente:

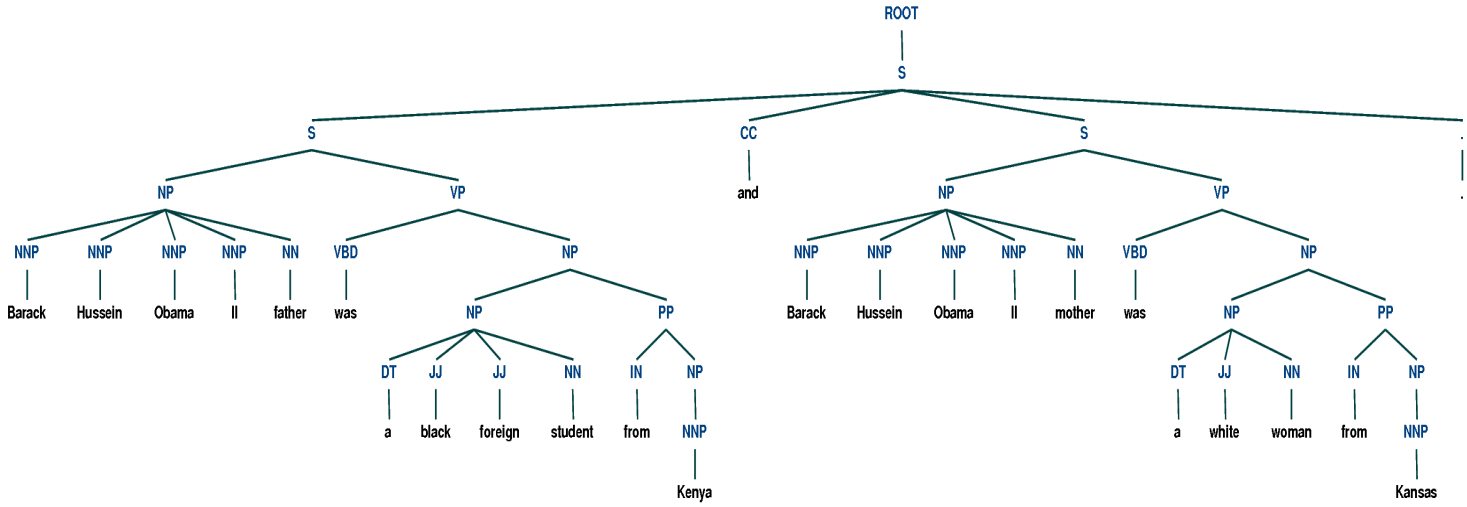


Figura 10: Exemplo de saída do Stanford parser para a sentença: *Barack Hussein Obama II father was a black foreign student from Kenya and Barack Hussein Obama II mother was a white woman from Kansas.*

-

5.5.2 Aproveitamento dos passos Anteriores

O parser aproveita o texto transformado pelo normalizador como se fosse o texto original, mas os termos trocados implicam em trocas nas classes morfológicas contidas em cada constituinte das sentenças, alterando as árvores geradas pelo parser.

A normalização das entidades e resolução dos pronomes fazem com que o parser sempre utilize as mesmas estruturas (NP, NNP) para posicionar as entidades nas árvores sintáticas. Isto não só diminui a complexidade das sentenças para o parser, consequentemente diminuindo a taxa de erros, como também diminui drasticamente a complexidade do próximo passo, a extração das relações.

5.6 Extrator de relações

O extrator de relações é o último passo do pipeline, e foi implementado no programa *extrai_rels_cstruct* e em seus módulos *grafos_rels* e *pseudo_tempo*, sendo a divisão feita da seguinte forma:

- *extrai_rels_cstruct*: Módulo principal. Cuida da entrada, processamento e saídas.
- *grafos_rels*: Representação interna do grafo de relacionamentos e suas transformações.
- *pseudo_tempo*: Representação do tempo e captura de expressões temporais.

Ele recebe como único argumento o nome de um arquivo XML produzido pelo Stanford parser. Também usa o arquivo de URIs gerado pelo normalizador para identificar NEs presentes nos sujeitos e objetos das orações.

5.6.1 Motivação

Para entender melhor o modo como nós, seres humanos, aprendemos e interpretamos nossa língua, o autor fez pequenos testes envolvendo perguntas e respostas à duas crianças em idade pré-escolar (entre 4 e 5 anos). A maneira como respondiam às perguntas, feitas a maioria na presença de ambos, denunciou uma busca por padrões predominantemente sintáticos e portanto de natureza estrutural, na hora de formular as respostas. A ordem das palavras na pergunta era essencial para a obtenção de alguma resposta, embora o vocabulário desconhecido fosse frequentemente contornado e problemas semânticos completamente ignorados. Nos exemplos abaixo, as estruturas possivelmente utilizadas para a interpretação foram destacadas entre colchetes:

- "O leite que você bebe, de onde vem?"
  (inversão da ordem típica da pergunta)
  Resultado: A pergunta não foi compreendida
- "[De onde] [vem] [o leite]?"
  Resultado A pergunta foi interpretada e respondida corretamente.
- "[Como] **matamos** o boi para [cortar a carne]?"
  (subordinação, vocabulário desconhecido em negrito)
  Resp: "Com a tesoura"
  Resultado: A pergunta foi respondida aproveitando apenas o vocabulário conhecido, ignorando o restante.
- "[Com quantos] bois se faz [um] bife?"
  (apesar de sintaticamente interpretável, a pergunta não faz sentido)
  Resultado: A pergunta foi prontamente respondida com um número, ignorando o problema semâmtico.

Estas observações superficiais mostraram que as hipóteses sobre a aquisição da linguagem pelo ser humano tratadas por Chomsky (Chomsky, 1986)[11] podem ser notadas sem grandes dificuldades, dando o impulso necessário ao uso de regras sintáticas como mecanismo para a extração de relações.

5.6.2 Funcionamento

O funcionamento do extrator é baseado em um conjunto de regras, que são acionadas de forma independente, em um sistema similar ao de tratamento de eventos. Uma regra define um conjunto de condições sob os quais deve ser acionada, um conjunto de ações que devem ser executadas caso as condições sejam atendidas, após as quais a regra termina, e um valor de retorno indicando sua aplicação ou não.

O extrator percorre o documento XML especificado em sua entrada, procurando pelas tags "<parse>" e interpretando o conteúdo de cada uma destas em uma representação interna da árvore sintática equivalente, sob a qual inicia uma busca em profundidade, tentando aplicar cada uma das regras a todos os nós percorridos na busca (figuras 13, 14, 16, 18, 20, 22, 23 e 25).

Representação do grafo de relacões

Para a representação do grafo, foi desenvolvido um modelo simples e flexível, que permitisse a transformação para RDF e outros formatos e que pudesse incluir novas informações sem precisar de alterações.
O modelo desenvolvido é apresentado na figura abaixo:

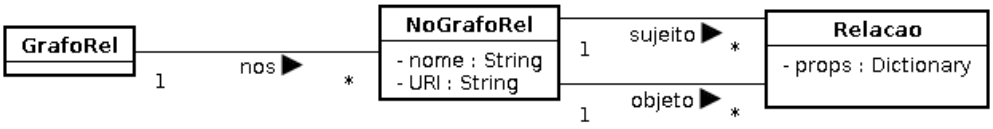


Figura 11: Modelo de dados para o grafo de relações. O atributo *props* guarda um conjunto extensível de informações sobre a relação.

Neste modelo, os elementos do grafo de relações são representados pelas classes *NoGrafoRel* (nó) e *Relação* (aresta). A classe GrafoRel serve de referência ao grafo como um todo. As reificações (arestas de linhas pontilhadas) são representadas pelo atributo *props*.

Este modelo permite a reificação das relações ao incluir um dicionário de informações no atributo *props*. A reificação é feita com o uso de chaves predeterminadas: "time" ou qualquer preposição, como "for" e "in" ("time" para as reificações em tempo e "for" para reificações em causa respectivamente).

Uma ilustração simples de reificação no tempo pode ser dada pela figura abaixo:

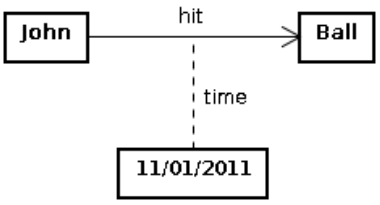


Figura 12: Exemplo de reificação: informação de tempo sobre um fato para a sentença: *John hit the Ball on November 1, 2011.*

Representação do tempo

Muitas das datas encontradas nos textos em linguagem natural são imprecisas, podendo ser divididas em:

- Explícitas: quando a data é escrita como uma combinação qualquer de dia, mês e ano. Ex: "Born in January 5".
- Implícitas: quando a data é escrita de forma relativa, através de advérbio ou oração adverbial. Ex: "It'll be tomorrow", "He was in Berlin when the wall came down".

Apenas o caso explícito é tratado pelo extrator, visto que o implícito pode ser considerado um caso complexo de correferência.

Este tipo de construção implica na necessidade de uma representação de tempo que não seja necessariamente precisa, então a solução adotada foi a seguinte:

- Todas as referências a tempo são representadas como intervalos.
- A resolução máxima é de um dia. Nada abaixo é considerado.
- Os intervalos para dia, mês e ano são independentes.
- Um intevalo aberto $(-\infty, x)$ é considerado todo o tempo até x e $(x, +\infty)$ como todo o tempo a partir de x.
- Há constantes especiais para representar passado e futuro indeterminados e o presente.

Esta representação permite que datas imprecisas sejam capturadas e que possam ser progressivamente compostas em datas mais precisas, a medida em que o texto aumente a precisão. Por exemplo: um texto contendo a sentença "... that's what happened by 2010.", terá neste ponto o tempo (?/?/2010 - ?/?/2010). Se mais adiante é encontrada a sentença "Until August, this was the situation.", o tempo neste ponto do texto será (?/?/2010 - ?/8/2010), considerando que não houve quebras no discurso que mudassem o ano.

As interrogações na representação indicam a porção desconhecida do intervalo de tempo. Por exemplo: (?/?/2010 - ?/8/2010) significa "em algum dia e mês do ano de 2010 até agosto de 2010", onde a primeira parte do intervalo

é sempre menor ou igual à segunda. Este é o recurso usado para capturar o contexto temporal expresso parcialmente nas sentenças. O que quer dizer que em praticamente todas as datas obtidas dos textos haverá porções desconhecidas (interrogações). Quando as duas partes do intervalo são iguais e estão totalmente preenchidas, a data é exata.

A composição de datas foi implementada de modo a sempre aumentar a precisão enquanto não houver conflito em um dos intervalos (dia, mês ou ano). No caso de conflito, o intervalo encontrado mais recentemente no texto é escolhido.

Regras

Início da oração Acionada quando a busca em profundidade encontra um nó indicando o início da sentença, de uma oração coordenada, ou de uma oração subordinada. Tem como objetivo marcar a posição da árvore sintática onde foi iniciada a oração, para que esta possa ser terminada pela regra de fim da oração.

Ao ser acionada, atualiza a variável de nível da oração, que indica em que nível (profundidade) da árvore sintática se encontra a oração.

Para a árvore sintática da primeira sentença obtida no exemplo do Stanford parser:

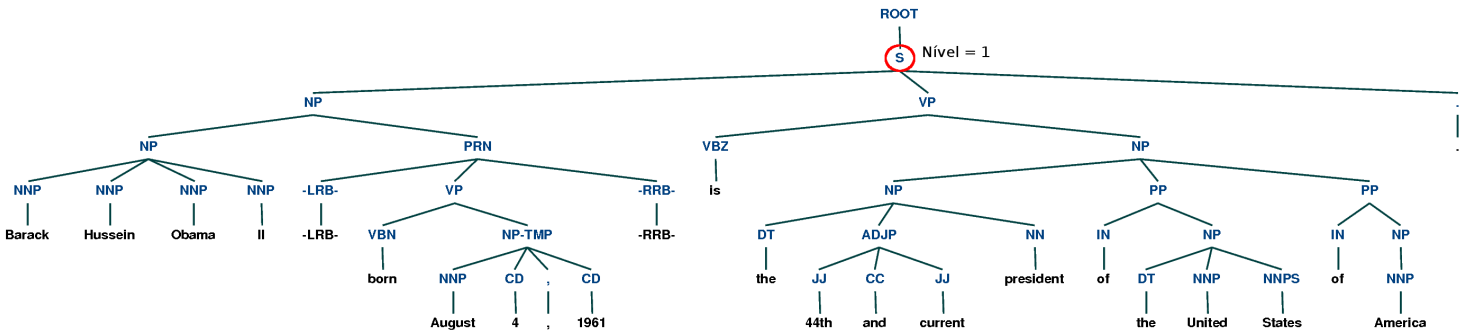


Figura 13: A regra de início da oração será acionada quando a busca alcançar os nós circulados.

Sujeito Acionada quando a busca encontra um nó indicando um sintagma nominal, diretamente abaixo do início de uma sentença ou oração, e que não possui nenhum SN[19] entre os filhos, indicando que se trata do SN de nível mais baixo (mais próximo as folhas) iniciando uma oração. Este é na maioria dos casos o sujeito da oração.

No escopo do trabalho, escapam a esta regra alguns casos com aposto. Tem como objetivo armazenar o sujeito para que possa ser posteriormente associado a um predicado.
Quando acionada, realiza as seguintes ações:

1. Concatena os nomes no caso de sujeito composto;

[19] Sintagma Nominal

2. Verifica se o sujeito é uma NE e atribui a ele uma URI em caso positivo;
3. Acrescenta o sujeito obtido como um nó do grafo de relações;
4. Coloca o sujeito em uma pilha de sujeitos;
5. Verifica a existência e captura datas explícitas no sujeito através da regra de tempo, como no caso "In September 2001, WTC was attacked".

Exemplo

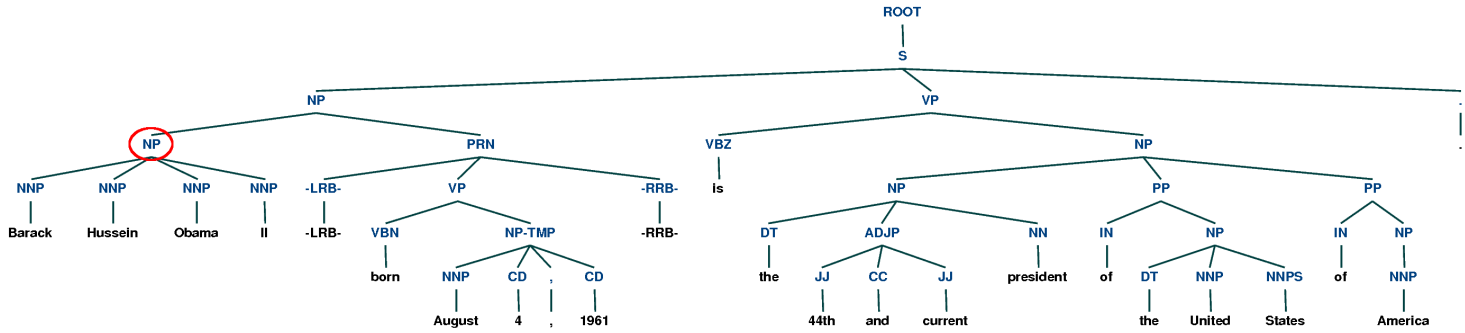


Figura 14: A regra de sujeito será acionada quando a busca alcançar o nó circulado.

O grafo para esta sentença ficará neste estado, após a aplicação da regra do sujeito:

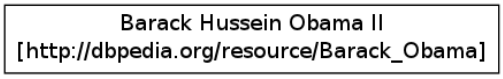


Figura 15: Grafo de relações após a aplicação da regra do sujeito, contendo apenas um nó com sua URI.

-

Predicado Acionada quando a busca encontra um nó indicando um verbo. Tem como objetivo associar objetos a um sujeito por meio de um predicado (o verbo).
Ao ser acionada, realiza as seguintes ações:

1. Verifica o tempo do verbo e aciona a regra de tempo para dar o tratamento adequado ao tempo verbal;
2. Concatena os verbos vizinhos no caso de locução verbal;
3. Coloca o verbo ou locução em uma pilha de verbos;

4. Acrescenta o texto do predicado (menos o verbo) como um nó no grafo de relações;

5. Verifica a existência e captura datas explícitas no predicado através da regra de tempo, como no caso *Iran has advanced its nuclear program in 2011*;

6. Acrescenta uma aresta no grafo de relações, ligando o sujeito no topo da pilha de sujeitos ao predicado recém encontrado. A aresta tem como nome o verbo no topo da pilha de verbos e tem atribuida a ela o tempo que está no topo da pilha de tempo (reificada no tempo).

Exemplo

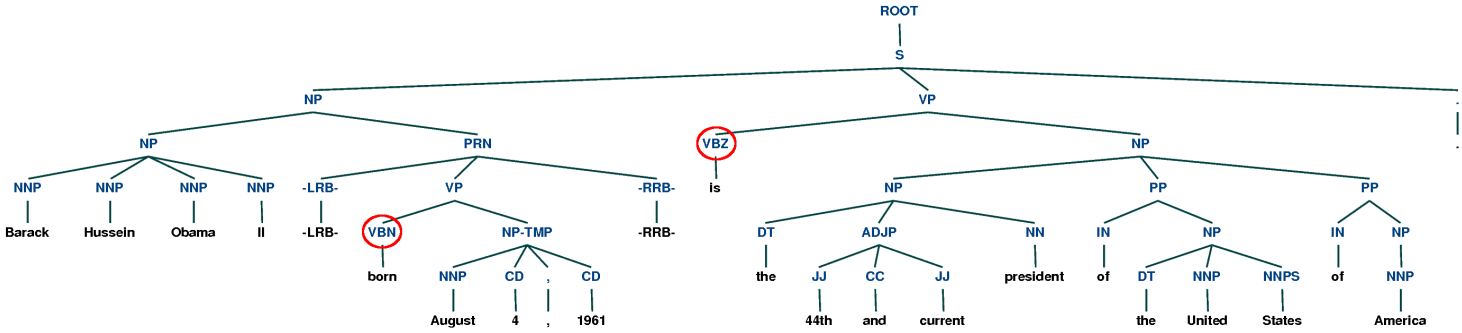


Figura 16: A regra de predicado será acionada quando a busca alcançar os nós circulados.

O grafo para esta sentença ficará neste estado, após a aplicação da regra do predicado:

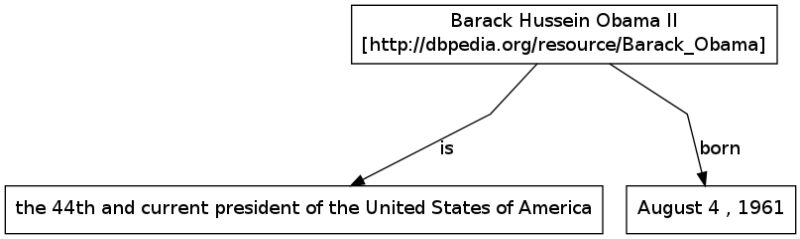


Figura 17: Grafo de relações após a aplicação da regra do predicado

-

Objeto, núcleo do objeto e predicativo do sujeito Acionada quando a busca encontra nó indicando um sintagma nominal que não possui nenhum SN como filho e a pilha de verbos não está vazia. Tem como objetivo identificar o

núcleo do objeto de um predicado, facilitando a ligação entre entidades nomeadas no grafo de relacionamentos.
Quando acionada, realiza as ações:

1. Concatena os nomes no caso de objeto composto;
2. Verifica se o objeto é uma NE e atribui a ele uma URI em caso positivo;
3. Atribui o objeto obtido como o "núcleo" da última relação acrescentada ao grafo.

Exemplo

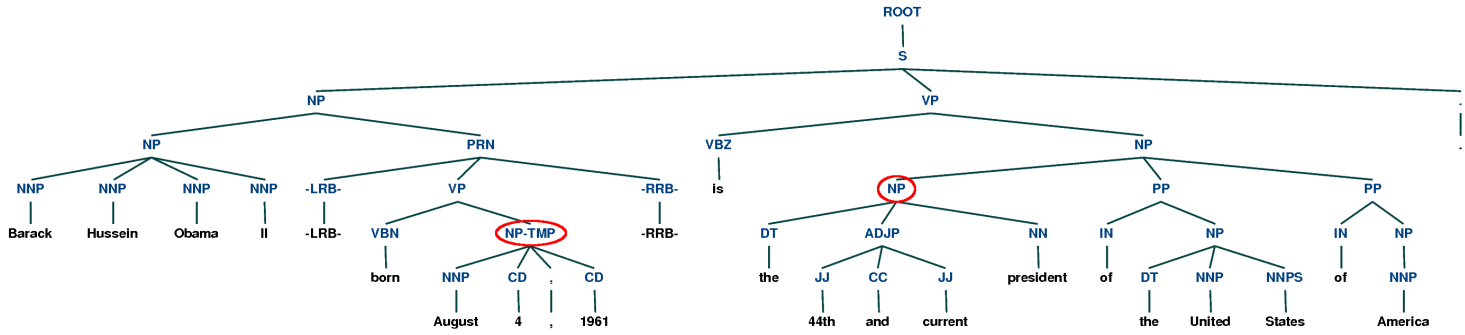


Figura 18: A regra do objeto e predicativo será acionada quando a busca alcançar os nós circulados.

O grafo para esta sentença ficará neste estado, após a aplicação da regra do objeto e predicativo:

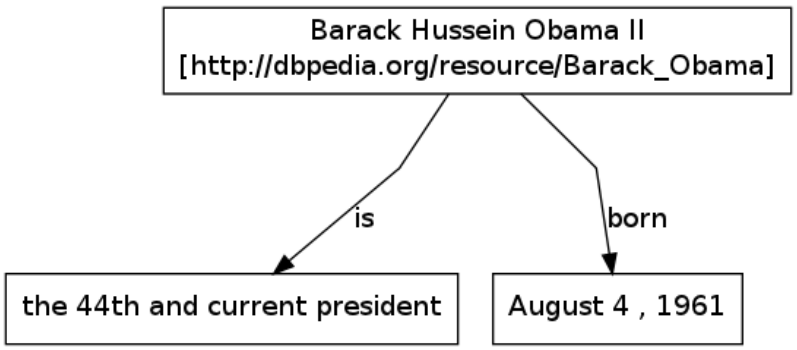


Figura 19: Grafo de relações após a aplicação da regra do objeto e predicativo. Apenas os núcleos são mantidos nos nós de objetos.

•

Complemento nominal Acionada quando a busca encontra um nó indicando um sintagma nominal, que não possui nenhum SN como filho e que possui um sintagma preposicionado entre seus irmãos. Tem como objetivo encontrar relações de posse e pertinência em sujeitos e objetos, ambos contidos em sintagmas nominais.
Quando acionada, realiza as ações:

1. Concatena o sintagma obtido e o acrescenta como um nó no grafo de relações;
2. Verifica a presença de sintagmas nominais vizinhos sendo ligados por preposição;
3. Acrescenta uma aresta no grafo de relações para cada um dos sintagmas obtidos na ação anterior, ligando-os ao primeiro sintagma obtido. A aresta terá como nome a preposição que liga so sintagmas e como tempo o elemento no topo da pilha de tempo.

Exemplo

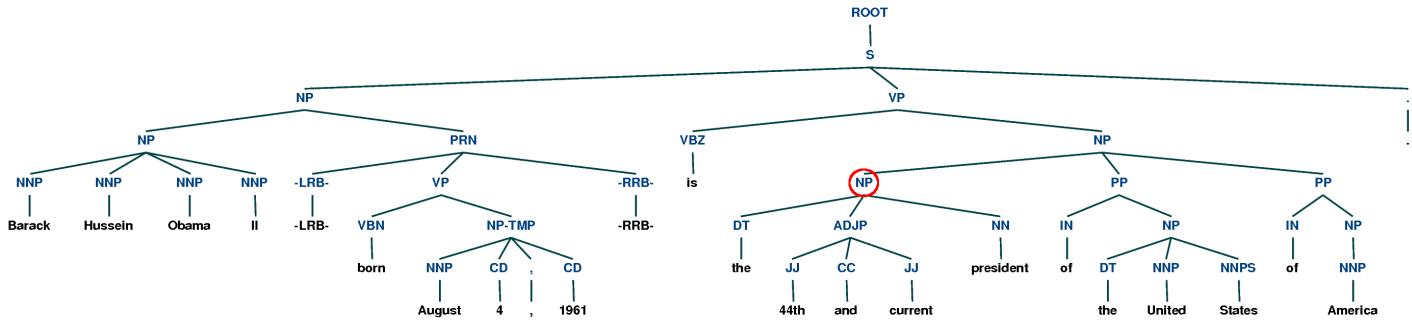


Figura 20: A regra de complemento nominal será acionada quando a busca alcançar os nós circulados.

O grafo para esta sentença ficará neste estado, após a aplicação da regra do complemento nominal:

•

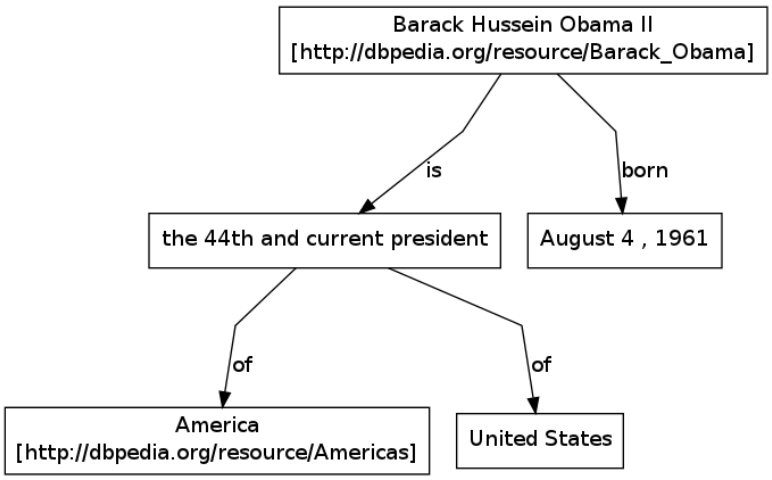


Figura 21: Grafo de relações após a aplicação da regra do complemento nominal. Os complementos são associados aos objetos.

Sintagma preposicionado Acionada quando a busca encontra um nó indicando uma preposição. Tem como objetivo encontrar modificadores causais e de lugar em predicados já obtidos. Ignora os sintagmas preposicionados que modificam nomes, que serão capturados pela regra de complemento nominal.

Como esta regra busca por modificadores de predicados, sua aplicação sempre resultará em uma reificação, expressando causa (eg. *for, to*) ou lugar(eg. *in, above*).
Ao ser acionada, realiza as ações:

1. Verifica se o sintagma já foi capturado pela regra de complemento nominal;
2. Concatena todo o sintagma, com exceção da preposição;
3. Verifica a existência e captura datas explícitas através da regra de tempo;
4. Atribui o sintagma obtido como "$PREP(i)$" da última relação acrescentada ao grafo, onde $PREP$ é a preposição encontrada pela regra e i é sua ordem de chegada. A ordem é necessária pois podem existir várias preposições iguais em uma mesma sentença, referindo-se ao mesmo sujeito.

Exemplo

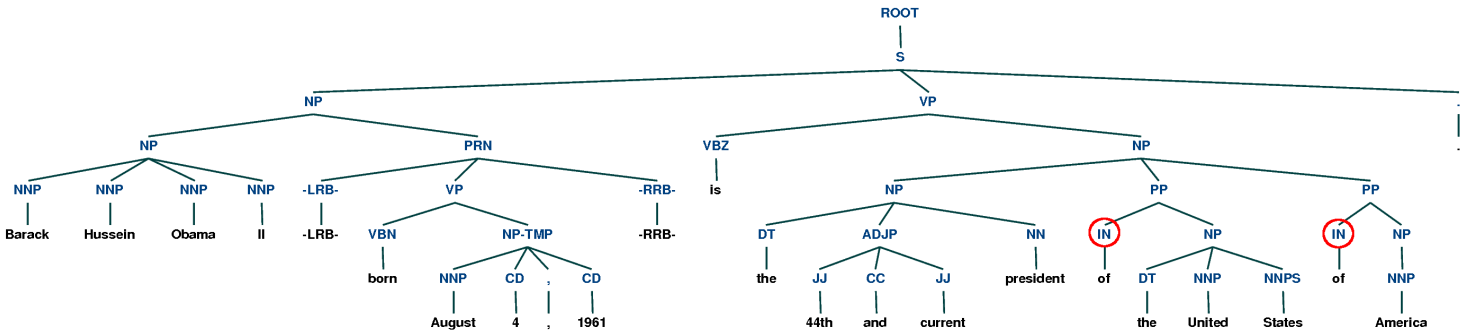


Figura 22: A regra de sintagma preposicionado será acionada quando a busca alcançar os nós circulados.

Nesta sentença, as preposições não estão modificando o predicado e portanto já foram capturadas pela regra de complemento nominal. Logo, o grafo não sofre alterações. Um exemplo de aplicação da regra é exibido na figura 26. •

Tempo Única regra que não é aplicada sobre nós da árvore sintática. É chamada por outras regras diretamente sobre o texto e é acionada quando o texto contém alguma data explícita, ou quando é um verbo. Tem como objetivo identificar o tempo onde ocorreram os predicados ou onde eles são válidos, permitindo agregar temporalidade ao grafo de relações.

As datas são detectadas por meio de um conjunto de expressões regulares, aplicadas em sequência sobre o texto, e os campos de um intervalo (dia, mes e ano) vão sendo preenchidos conforme o casamento ou não das expressões.
Ao ser acionada, realiza as ações:

1. Preenche uma instância de tempo com a data ou período detectado;
2. Faz a composição do tempo preenchido com todos os elementos da pilha de tempo (seção 5.6.2);
3. Substitui em todos os predicados da sentença atual os tempos indeterminados (presente e passado ou futuro indeterminados) pelo tempo preenchido.

Exemplo

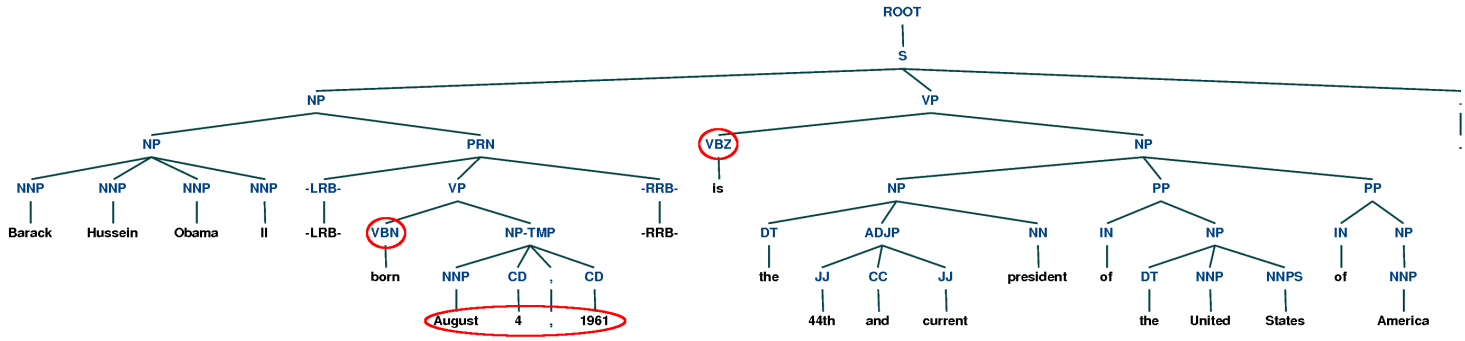


Figura 23: A regra de tempo será acionada quando a busca alcançar os nós circulados.

O grafo para esta sentença ficará neste estado, após a aplicação da regra do tempo:

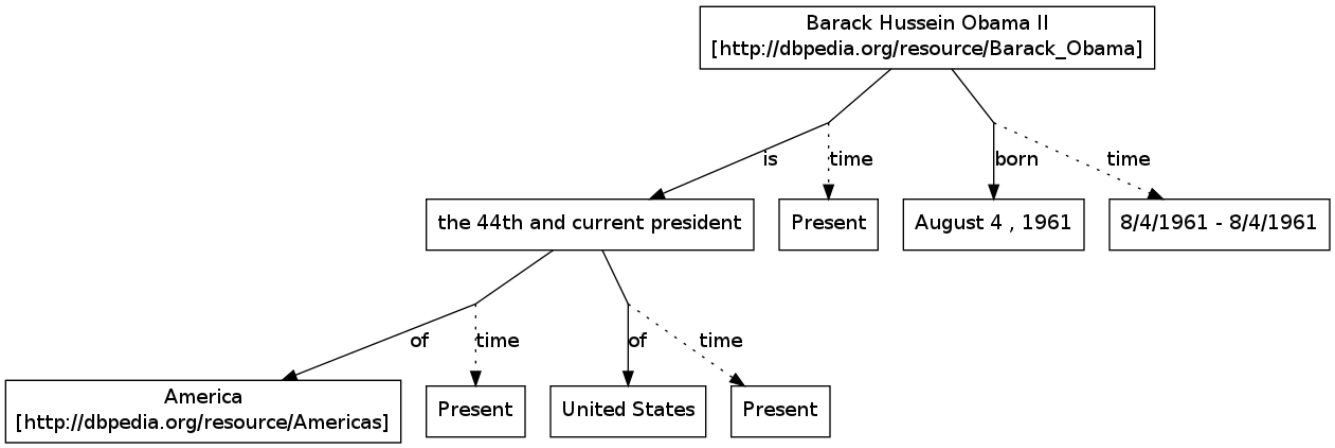


Figura 24: Grafo de relações após a aplicação da regra do tempo. Um tempo é atribuido a cada predicado no grafo.

-

Fim da oração Acionada quando a profundidade atual da busca está acima do nível da oração atual. Tem como objetivo terminar a oração, retornando à posição correta na hierarquia sintática da sentença.
Quando acionada, realiza as ações:

1. Descarta o topo das pilhas de sujeito, verbo e tempo;

2. Atualiza o nível da oração atual.

Exemplo

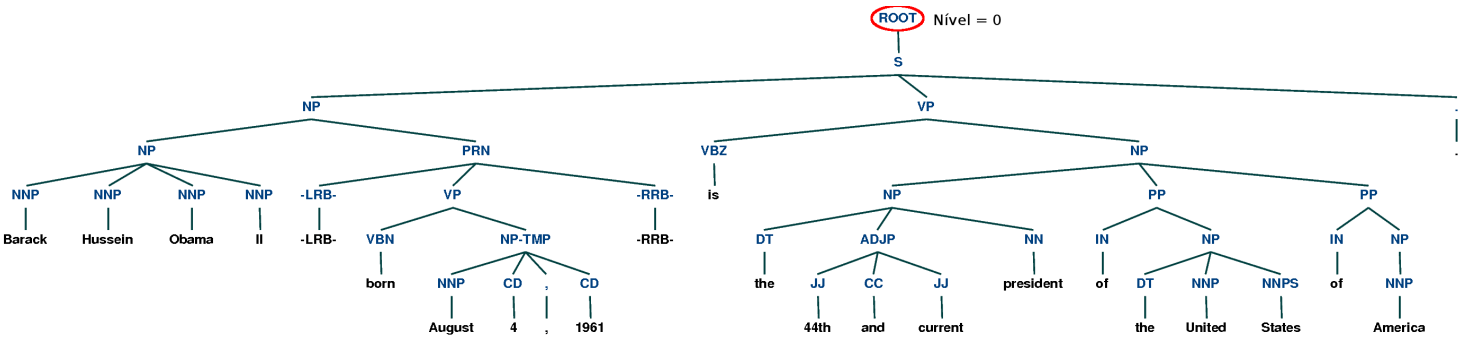


Figura 25: A regra de fim da oração será acionada quando a busca alcançar o nó circulado.

A aplicação da regra de fim da oração não modifica o grafo de relações.

Abaixo os grafos para a segunda e terceira sentenças do exemplo do Stanford parser:

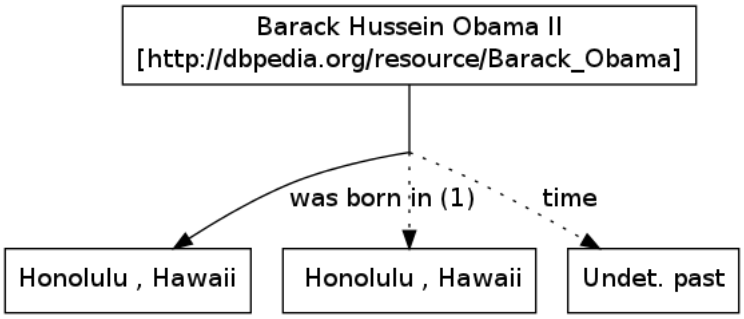


Figura 26: Grafo extraído para a sentença *Barack Hussein Obama II was born in Honolulu, Hawaii.*

•

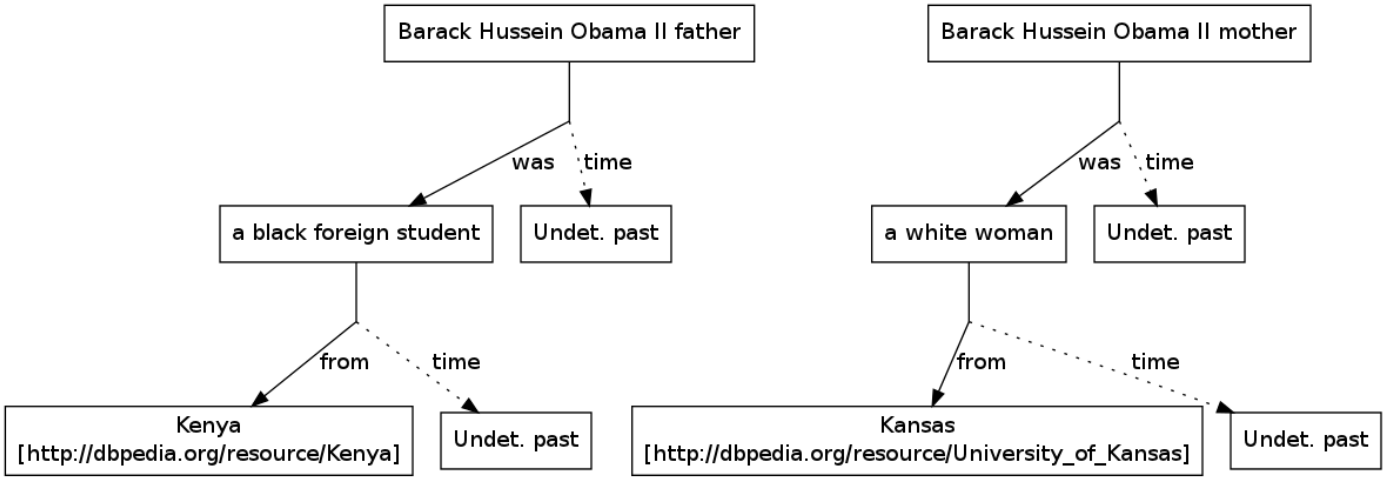


Figura 27: Grafo extraido para a sentença *Barack Hussein Obama II father was a black foreign student from Kenya and Barack Hussein Obama II mother was a white woman from Kansas.*

5.6.3 Saída

O extrator gera três saídas:

- O grafo de relações para todo o texto, em formato DOT (GraphViz, 2004)[22].
- Grafos de relações para cada sentença do texto, também em formato DOT.
- As árvores sintáticas para cada sentença, em formato PostScript (PS).

Os grafos em formato DOT podem ser convertidos em um formato popular de imagens, como o JPEG ou PNG, para visualização.

5.6.4 Aproveitamento dos passos Anteriores

O extrator utiliza a saída do Stanford parser diretamente, interpretando a notação de árvores sintáticas no formato *Penn Treebank* através de rotinas da biblioteca NLTK (Bird, Klein, Loper, 2009)[6]. A representação interna das árvores obtida desta forma possui desempenho ótimo para buscas.

A normalização de entidades e resolução de correferências faz com que todas as NEs apareçam como substantivos em sintagmas nominais nas árvores sintáticas, em vez de aparecerem como pronomes ou orações substantivas por exemplo. Isto permite que sejam capturadas com apenas duas regras (sujeito e compl. nominal).

O arquivo de URIs gerado pelo normalizador também é aproveitado para identificar as entidades nomeadas no grafo de relações e enriquecê-lo com links para a ontologia (DBpedia) nos nós contendo NEs.

5.7 Testes preliminares

Para avaliar o desempenho do pipeline como um todo, foi feita uma comparação inicial com a ferramenta de mesma função que havia sido testada, o ReVerb (Fader, Soderland, Etzioni, 2011)[15].

Os textos foram os mesmos escolhidos para avaliar o ReVerb: Os artigos da *Simple English Wikipedia* sobre Barack Obama e New York.
Após a comparação ficaram as seguintes impressões:

- Relações curtas (verbais) extraídas pelo Reverb eram em boa parte extraídas também pelo pipeline.
- Um grande número de relações curtas não extraídas pelo ReVerb foram extraídas corretamente pelo pipeline.

Estes resultados estavam de acordo com a expectativa de que o pipeline extrairia mais relações curtas (verbais), visto que o ReVerb possui uma tendência a fazer extrações longas (expressões). Entretanto, a abrangência (recall) do pipeline ficou melhor que o previsto.

Isto melhorou a perspectiva sobre o funcionamento do pipeline, e direcionou o trabalho para a melhoria da precisão do extrator, que ainda estava em estágio de protótipo, de acordo com a estratégia de melhor esforço.

Posteriormente foi acrescentado aos testes o artigo sobre a General Eletric da *English Wikipedia*, como uma forma de verificar o efeito do aumento da complexidade dos textos sobre o pipeline. Os resultados foram similares aos dos outros textos e melhoraram de forma consistente ao longo do trabalho.

5.8 Trabalhos Relacionados

Harrington e Clark (Harrington, Clark, 2008)[23] descrevem o AskNet, um sistema de extração de informação capaz de construir redes semânticas de larga escala a partir de textos não extruturados. O pipeline de extração começa pelo parsing das sentenças no texto usando o parser C&C, um parser baseado no formalismo linguístico da Gramática Categórica Combinatória (CCG[20]) (Steedman, 1987)[36]. Um estágio de Reconhecimento de Entidades Nomeadas (NER) é feito usando o anotador C&C NER, que é um anotador do tipo *entropia máxima* (Jaynes, 1963)[24].

Depois que as sentenças são interpretadas, o AskNet usa a ferramenta *Boxer* para análise semântica (Curran, Clark, Bos, 2007)[14], a qual produz uma representação em lógica de primeira ordem baseada na semântica da Teoria de Representação de Discurso (DRT[21]) (Kamp, 1981)[25].

Atualizações na rede semântica com novos grafos (derivados de sentenças) é feita pela aplicação da técnica "spreading activation" (Nilsson, 1998, p. 121-122)[32] para alinhar as entidades da nova estrutura às entidades na rede maior. Uma abordagem de baixa cobertura para resolução de pronomes é usada para as correferências pronominais.

Harrington e Clark (Harrington, Clark, 2008)[23] também propõem uma metodologia para avaliar as redes semânticas de larga escala. A avaliação deste

[20] Combinatory Categorial Grammar
[21] Discourse Representation Theory

trabalho é uma adaptação da técnica de avaliação proposta por eles, usando a Wikipedia como corpus e um conjunto similar, embora diferente, de tipos de erros. Este trabalho é inspirado e relacionado ao trabalho de Harrington e Clark.

Mas apesar de ter objetivos similares, a abordagem usada no pipeline de extração é significativamente diferente quanto à estratégia de parsing, NER e resolução de correferências pronominais. Esta foi feita visando um cenário de representação de grafos simplificados, onde a representação não é mediada pela DRT. As outras diferenças maiores são o uso da Wikipedia como corpus e o foco em maximizar a compatibilidade com o modelo RDF.

Wojtinnek et al. (Wojtinnek et al., 2010)[49] propõe uma extensão do AskNet que oferece uma tradução da saída do AskNet para RDF. O trabalho de Wojtinnek et al. não provê resultados experimentais que possam ser utilizados para a avaliação da abordagem de extração.

6. Experimentos

Os experimentos tiveram como foco a verificação do desempenho do pipeline como um todo. Portanto, foi necessária a utilização de textos com características variadas em termos de vocabulário e sintaxe, para uma visão minimamente abrangente da língua.

Os textos utilizados foram processados pelo pipeline e em seguida avaliados quanto a corretude em um conjunto de aspectos, que serão abordados na seção 6.2.

6.1 Obtendo o texto

6.1.1 Wikipedia: English X Simple English

O uso da Wikipedia como fonte dos textos para o trabalho já havia sido planejado antes mesmo do seu início, visto que a Wikipedia é a maior fonte de informação organizada existente na internet nos dias atuais. Tendo em vista o objetivo do trabalho, que é extrair fatos do texto, e a intenção prévia de aproveitar e complementar os dados estruturados da DBpedia, a Wikipedia tornou-se a escolha natural.

Além disto, os esforços de extração de informações para estruturação na DBpedia têm sido voltados em grande parte para a língua inglesa, novamente direcionando o trabalho para esta língua.

Entretanto, a Wikipedia da lingua inglesa é dividida em duas versões paralelas, relativas ao tratamento da língua: a *English Wikipedia* e a *Simple English Wikipedia*. A primeira é a versão que deu origem ao projeto e ainda hoje é a mais utilizada e editada dentre todas as línguas. A segunda foi criada com o objetivo de melhor passar as informações aos leitores não nativos da língua inglesa.

A principal diferença está no vocabulário e tipos de construção sintática utilizados, que tendem a ser mais simples na versão "Simple". Isto levou à utilização de ambas no trabalho para avaliar o impacto causado pela diferença de complexidade sintática nas extrações.

O resultado esperado era de que as extrações seriam mais precisas na versão "Simple", refletindo a redução da complexidade do texto. Contudo, o contrário ocorreu em muitos casos, indicando que mesmo sentenças simples do ponto de vista estrutural podem ser de análise difícil por utilizarem uma ordenação incomum dos termos, ou por estarem fora da norma culta. Isto é importante, considerando que as regras de extração produzidas para o trabalho cobrem um conjunto reduzido de padrões sintáticos da língua.

6.1.2 Obtendo e separando os artigos

O texto de interesse para este trabalho está contido nos artigos da Wikipedia, que podem ser acessados através de URIs (ex: http://en.wikipedia.org/wiki/USA), na forma de documentos HTML.

Entretanto, nesta forma o texto está misturado com marcações HTML, incluindo elementos não textuais como cabeçalhos e tabelas, que por não fazerem parte da língua, se comportam como "ruído" em todos os passos do pipeline e precisam ser removidos antes qualquer processamento. Isto conflita com a solução utilizada popularmente para obter textos da internet, que é percorrer as páginas usando um crawler[22]. Mas no caso da Wikipedia, há uma forma melhor de obter acesso aos textos: o *Wikipedia Database Dump*.

Wikipedia Database Dump

Trata-se de uma cópia integral do banco de dados da Wikipedia, condensado em um arquivo XML. Esta cópia é atualizada mensalmente e é feita em separado para cada recurso da Wikipedia em cada língua disponível. Assim, há arquivos para os dicionários, figuras e outros, incluindo os artigos. Neste trabalho foram utilizados os arquivos de artigos para "English" e "Simple English". Obtidos os arquivos, resta extrair o texto de cada um dos artigos, de forma que sejam utilizáveis pelo pipeline.

WikiExtractor

O texto dos artigos contido no arquivo condensado ainda contém marcações, que são aquelas feitas pelos seus autores, para destacar termos ou inserir e posicionar figuras por exemplo.

Para eliminar estas marcações foi criado o *WikiExtractor* (UNIPI/Medialab, 2009)[40], um programa que toma como entrada o arquivo XML condensado e remove todas as marcações, bem como todos os outros metadados do arquivo. O resultado é um novo arquivo XML contendo apenas os textos limpos, identificados por suas URLs da Wikipedia.

O *WikiExtractor* foi criado com a intenção de obter um corpus englobando todo o conteúdo dos artigos da Wikipedia, inicialmente em sua versão italiana. Como as marcações nos textos independem da língua, ele também pode ser usado nas versões de língua inglesa. Sua eficácia garantiu a ausência de ruídos nos textos utilizados neste trabalho.

Separador de artigos

O *WikiExtractor* realiza a limpeza do texto, mas não separa os artigos em arquivos individuais, o que é necessário para a seleção dos artigos que serão usados nos experimentos.

Para esta tarefa foi desenvolvido um separador de artigos como parte do trabalho. Ele usa as marcações de início e fim dos artigos produzidos pelo *WikiExtractor* para separar os textos e os identificadores (URLs) para gerar nomes únicos para os arquivos.

O separador foi implementado no programa *separa_artigos_wikiextractor* e recebe como argumentos o nome do arquivo produzido pelo *WikiExtractor* e o nome do diretório onde ficarão os artigos separados, cada um com o nome de seu identificador na Wikipedia (ex: http://en.wikipedia.org/wiki/New_York -> New_York).

[22] Programa que navega pelas páginas da Web de forma automática, resolvendo os links encontrados em cada página.

6.1.3 Seleção dos artigos

Para atender a um critério mínimo de confiabilidade para os experimentos, buscando aproximar-se de (Harrington, Clark, 2008)[23], foram selecionados cinco artigos dentre todos da Wikipedia. Eles foram selecionados de forma aleatória, atendendo aos seguintes critérios:

- 2 artigos sobre pessoas.
- 2 artigos sobre organizações.
- 1 artigo sobre um lugar.
- Conteúdo $>$ 40K caracteres para "English", $>$ 2K para "Simple English".

Para este fim, foi desenvolvido o programa *seletor*, que utiliza o serviço de queries *SparQL*[23] (W3C, 2008)[48] da DBpedia para obter uma lista com entidades dos tipos desejados. O *seletor* realiza o procedimento descrito abaixo para cada tipo de entidade desejada:

1. Consulta a DBpedia por todas as entidades de um tipo (PERSON, ORGANISATION, PLACE);
2. Sorteia uma entidade da lista obtida pela consulta;
3. Verifica se há artigos sobre a entidade sorteada nas duas versões da Wikipedia ("English" e "Simple English");
4. Verifica se ambos os artigos atendem ao critério de tamanho mínimo;
5. Caso as duas verificações sejam bem sucedidas, seleciona os artigos e remove-os da lista;
6. Retorna ao passo 2.

Este procedimento é feito duas vezes para o tipo pessoa (PERSON), duas vezes para o tipo organização (ORGANIZATION) e uma vez para o tipo lugar (PLACE).

6.2 Avaliação do desempenho

Após selecionados, os textos foram processados pelo pipeline, e os grafos resultantes foram avaliados em relação às sentenças já normalizadas das quais originaram, como no exemplo:

[23] Acrônimo recursivo para *SPARQL Protocol and RDF Query Language*.

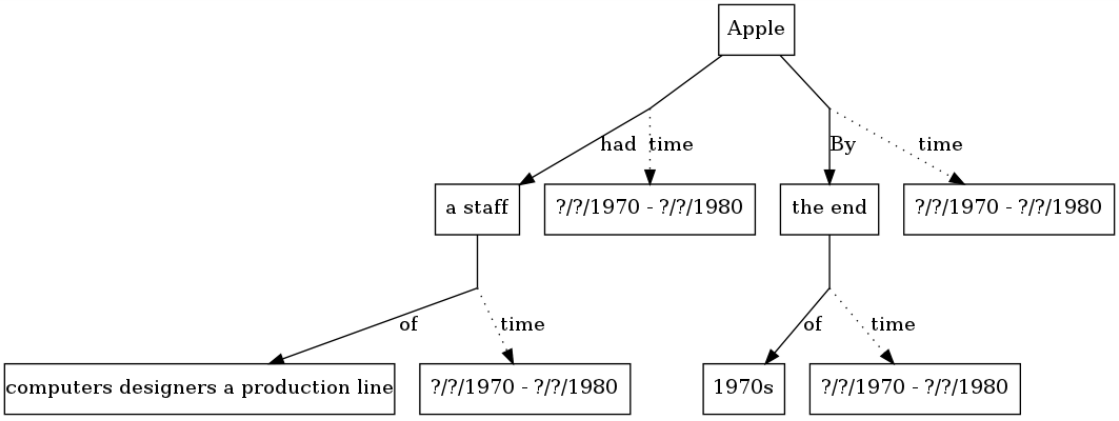


Figura 29: Grafo extraido para a sentença normalizada *By the end of the 1970s , Apple had a staff of computers designers and a production line.*

A avaliação consistiu da verificação da presença ou ausência de um determinado conjunto de erros na contrução do grafo, e de um determinado conjunto de atributos (features) da sentença.

6.2.1 Critérios da avaliação

Categorias de erros

As categorias de erros considerados na avaliação foram:

Resolução de correferência É o erro em resolver uma correferência pronominal, indicando tanto uma resolução incorreta quanto a não resolução. Apenas os pronomes pessoais retos e oblíquos, e pronomes possessivos são resolvidos e portanto somente estes são considerados.

Um sintoma típico deste erro é a presença de pronomes pessoais na sentença. Idealmente todos estes pronomes teriam sido substituidos por nomes após a execução do normalizador no pipeline. Abaixo, dois exemplos deste erro:

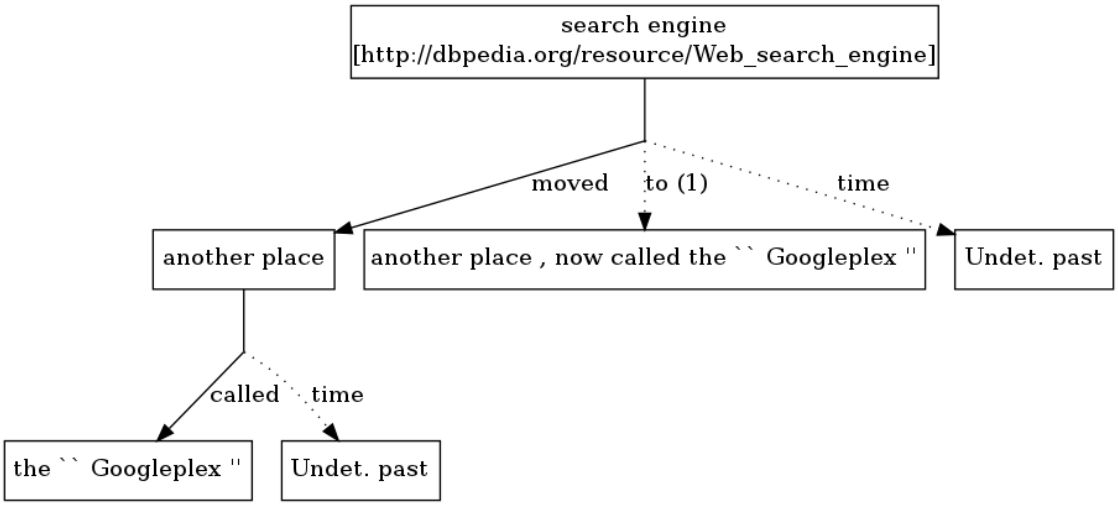


Figura 30: Na sentença *Then , later that year , search engine moved to another place , now called the " Googleplex " .* o termo "They" (Google) foi incorretamente substituido por "search engine".

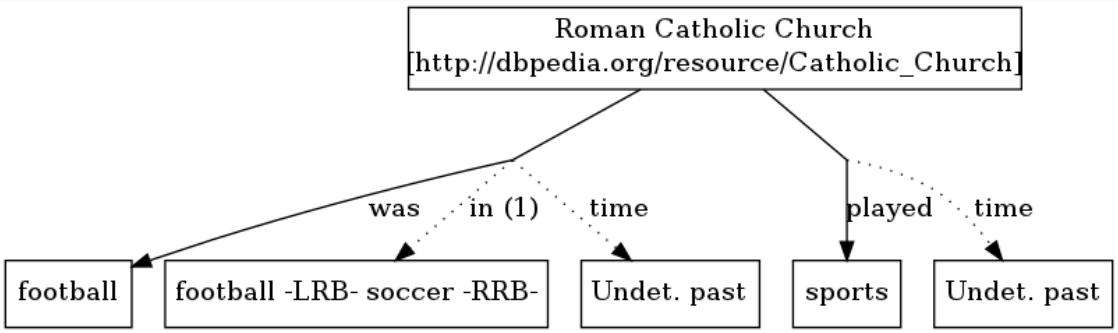


Figura 31: Na sentença *Roman Catholic Church played sports and was particularly interested in football (soccer) as a goalkeeper .* o termo "He" foi incorretamente substituido por "Roman Catholic Church".

•

Resolução de entidade nomeada Erro em mapear uma entidade nomeada a uma URI. Inclui tanto a ausência de URI para uma NE quanto a atribuição de uma URI errada. Sintomas típicos incluem a exibição de uma URI em um nó do grafo cujo nome não corresponde a esta URI e a não exibição de uma URI em um nó cujo nome é bem conhecido. Dois exemplos são dados abaixo:

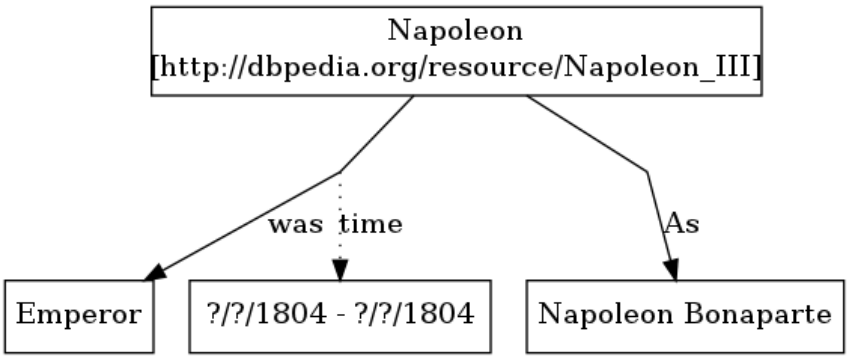


Figura 32: Na sentença *As Napoleon Bonaparte, Napoleon was Emperor of the French from 1804 to 1815. Napoleon* foi incorretamente resolvido como *Napoleon III* quando o correto seria *Napoleon I.*

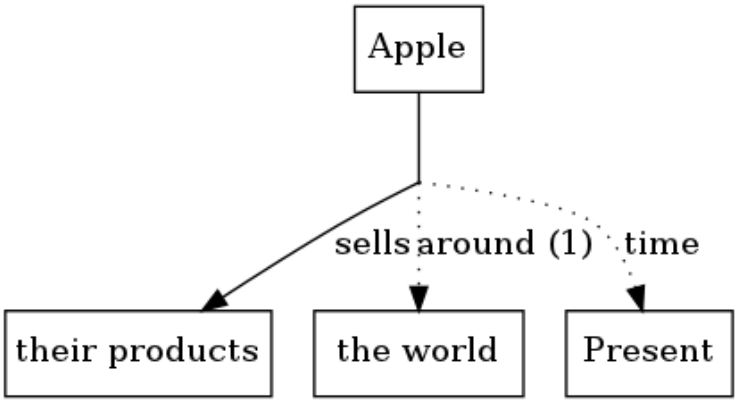


Figura 33: Na sentença *Apple sells their products all around the world.* o nome "Apple" não é resolvido em sua entidade correspondente (Apple Inc.).

•

Resolução de referência temporal explícita Erro ao associar uma data explícita a uma tripla. Inclui a falta da data, associação a uma tripla errada ou identificação errada da data. Dois exemplos abaixo:

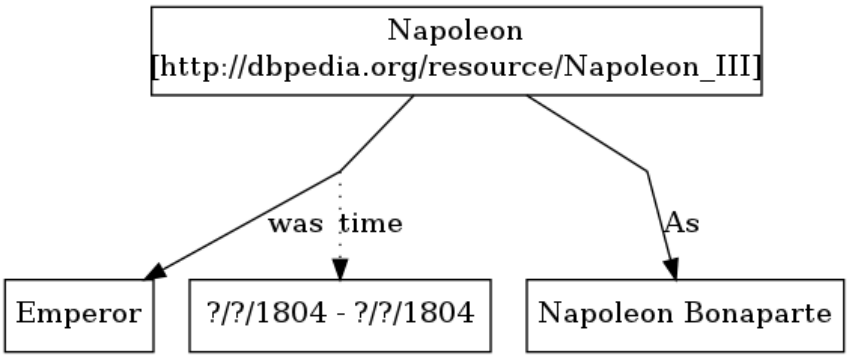


Figura 34: Na sentença *As Napoleon Bonaparte , Napoleon was Emperor of the French from 1804 to 1815.* Apenas o início do intervalo 1804-1815 foi resolvido.

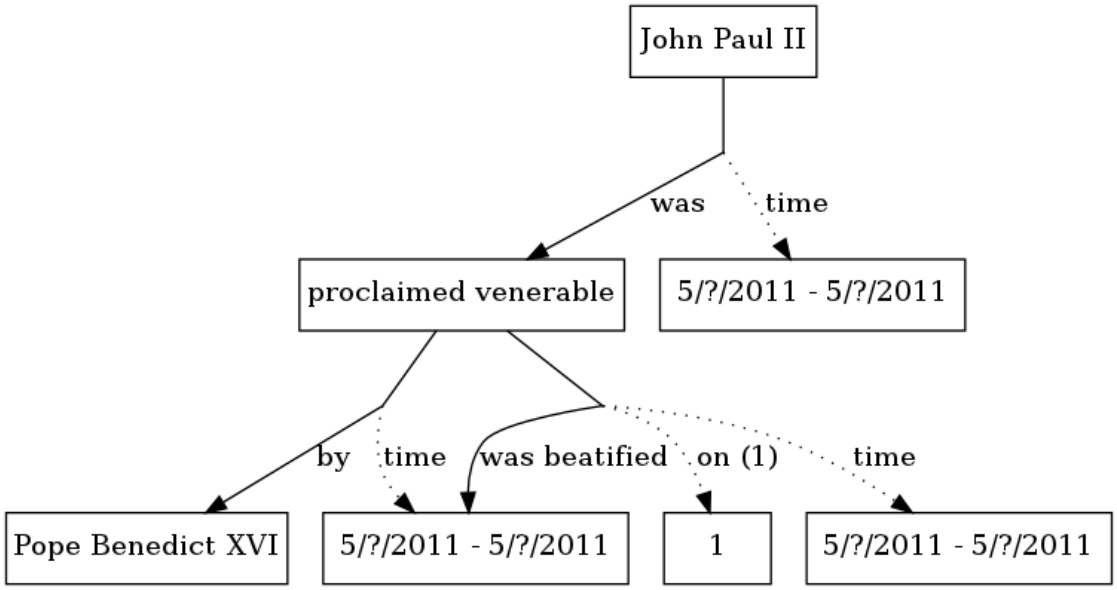


Figura 35: Na sentença *On 19 December 2009 , John Paul II was proclaimed venerable by German successor Pope Benedict XVI and was beatified on 1 May 2011 .* A primeira data foi perdida e a segunda extraida de forma incompleta.

-

Construção do sujeito Erro ao extrair o sujeito de uma tripla. Inclui o caso da exclusão de termos ou inclusão de termos que não fazem parte do sujeito, dentro do sujeito da tripla. O sintoma típico deste erro é a presença de palavras que não são nomes, artigos, pronomes ou preposições no sujeito. Dois exemplos abaixo:

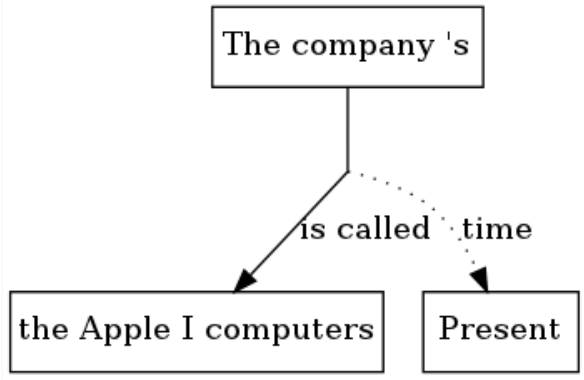


Figura 36: Na sentença *The company 's first product is now called the Apple I computers.* parte do sujeito "The company's first product" foi perdida.

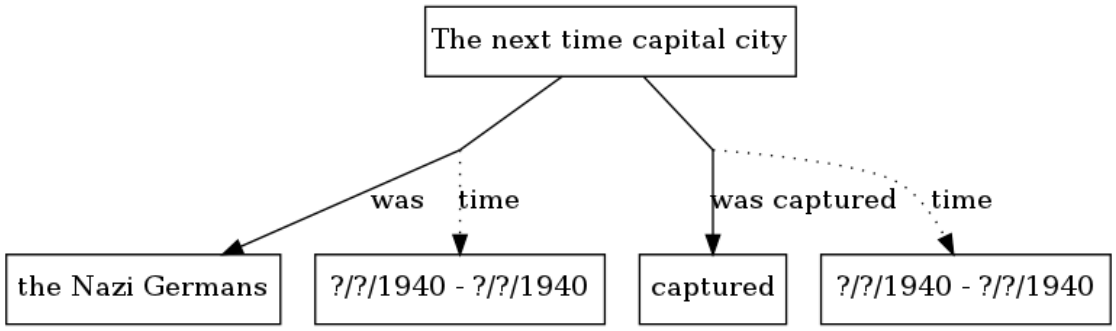


Figura 37: Na sentença *The next time capital city was captured was by the Nazi Germans in 1940.* a parte "capital city" não faz parte do sujeito.

-

Construção do predicado Erro ao extrair o predicado de uma tripla. Além de incluir os casos de exclusão de termos e inclusão de termos que não fazem parte do predicado, também inclui a falha em conectar ao sujeito ou objeto apropriado. Dois exemplos abaixo:

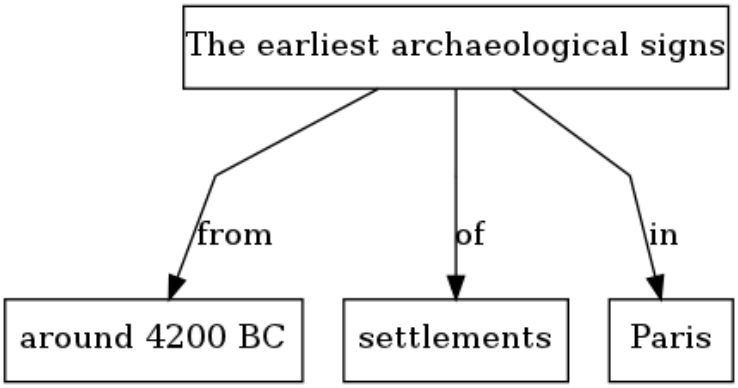


Figura 38: Na sentença *The earliest archaeological signs of permanent settlements in the Paris area date from around 4200 BC .* o predicado *date* não foi extraído.

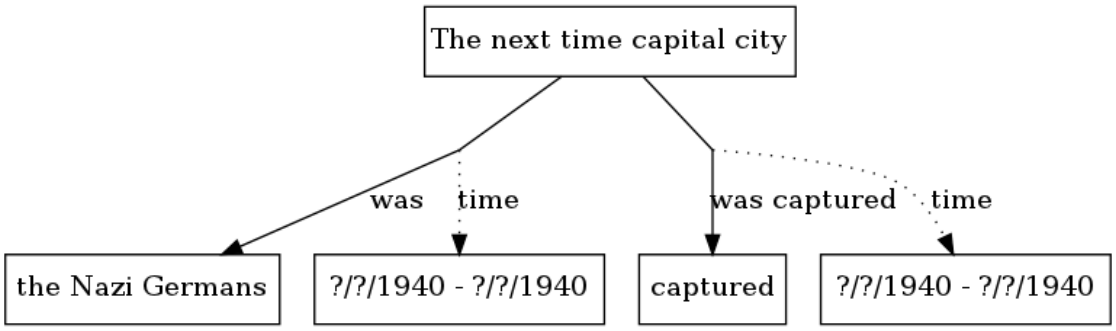


Figura 39: Na sentença *The next time capital city was captured was by the Nazi Germans in 1940.* a locução verbal não foi extraida corretamente.

•

Construção do objeto Erro ao extrair o objeto de uma tripla. Assim como no caso do sujeito, a ausência de termos do objeto ou inclusão de termos que não são do objeto estão inclusos nesta categoria. Dois exemplos abaixo:

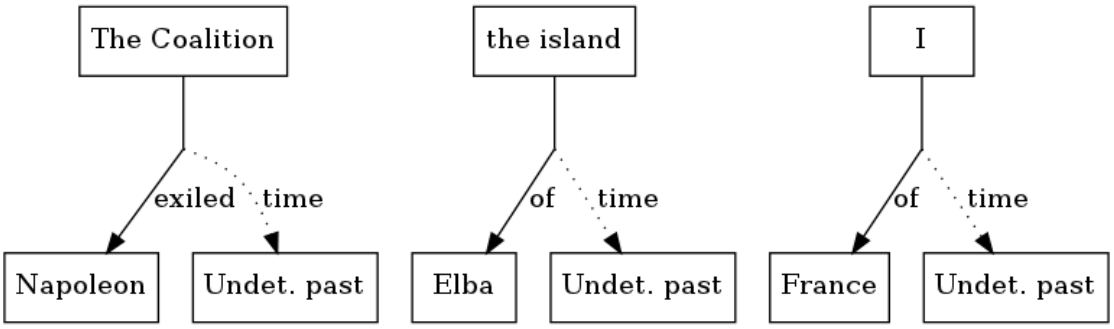


Figura 40: Na sentença *The Coalition also exiled Napoleon I of France to the island of Elba.* o objeto *Napoleon I of France* não foi extraido corretamente

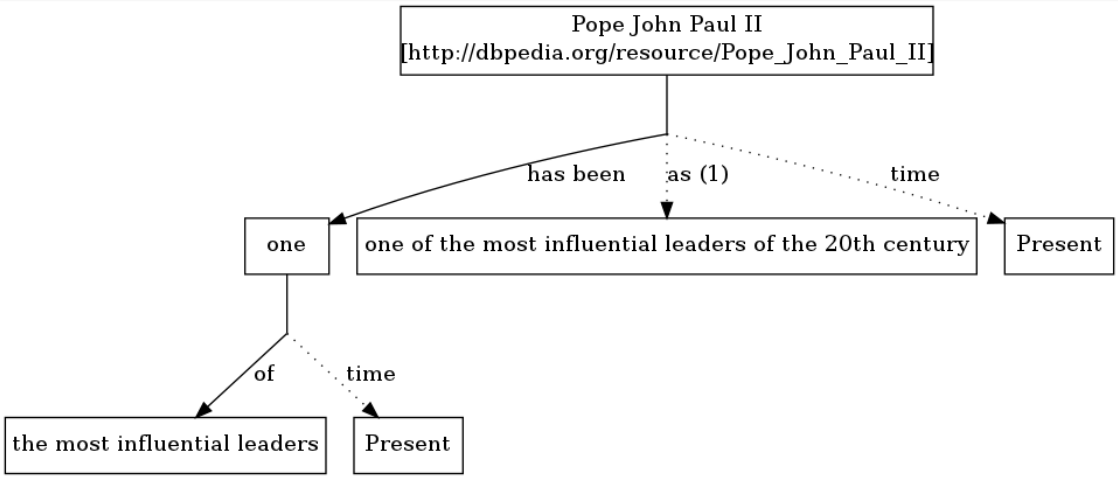


Figura 41: Na sentença *Pope John Paul II has been acclaimed as one of the most influential leaders of the 20th century.* o complemento nominal (*of the 20th century*) do objeto foi perdido.

-

Construção de reificação Erro ao atribuir um fato a uma tripla. Inclui a não atribuição de um fato sobre a tripla que esteja contido na sentença, a atribuição de um fato errado, ou a atribuição de um fato correto à tripla errada. Os erros envolvendo reificação no tempo, não entram nesta categoria, sendo classificados como erros de resolução de referência temporal explícita. Dois exemplos são exibidos abaixo:

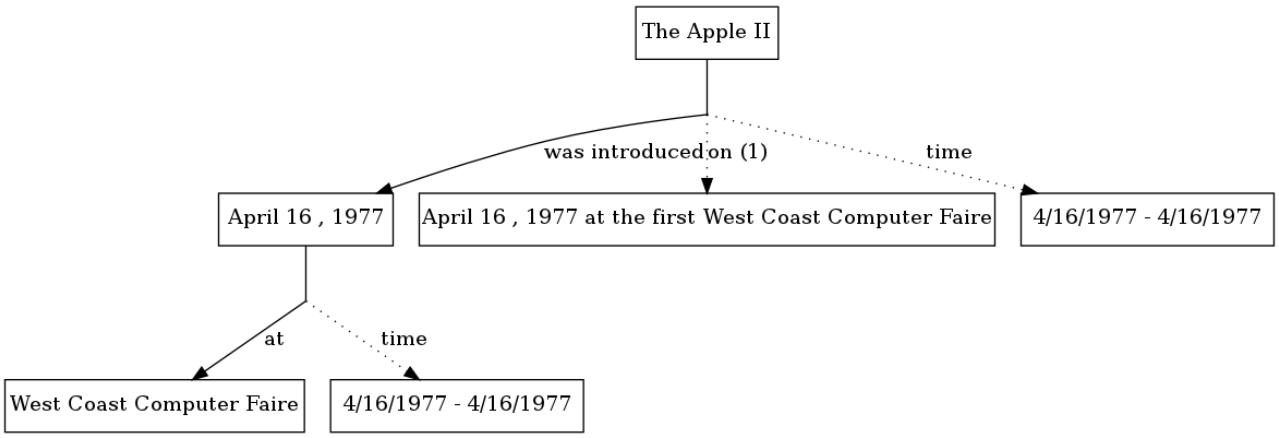


Figura 42: Na sentença *The Apple II was introduced on April 16 , 1977 at the first West Coast Computer Faire .*

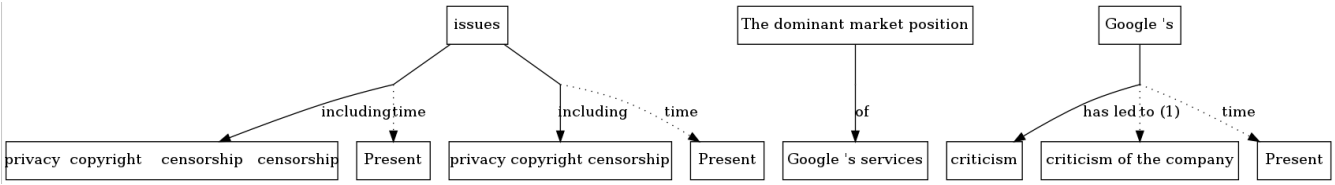


Figura 43: Na sentença *The dominant market position of Google 's services has led to criticism of the company over issues including privacy , copyright , and censorship .* o modificador "over" não foi reificado

•

Construção do caminho de triplas Erro ao construir uma ligação entre triplas (caminho de triplas). Inclui a não ligação de triplas que deveriam estar ligadas e a ligação de triplas usando palavras que não são conectores. Um sintoma típico deste erro é a construção de um grafo desconexo para uma sentença sem coordenação ou subordinação. Abaixo estão dois exemplos:

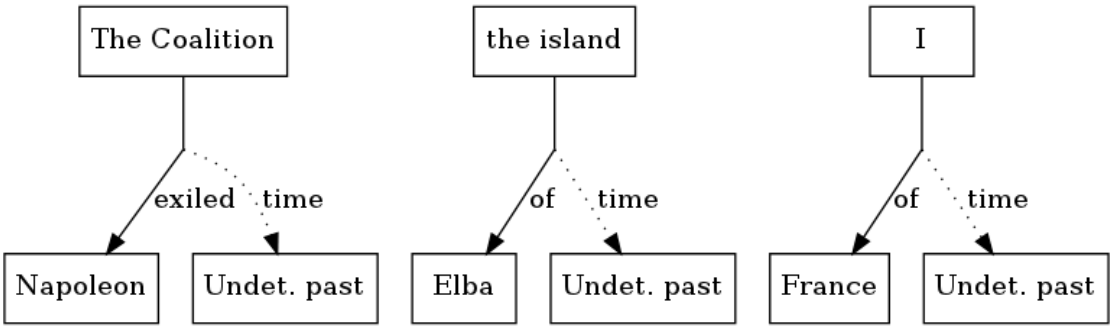


Figura 44: Na sentença *The Coalition also exiled Napoleon I of France to the island of Elba.* o grafo deveria ser conexo pois há apenas uma oração.

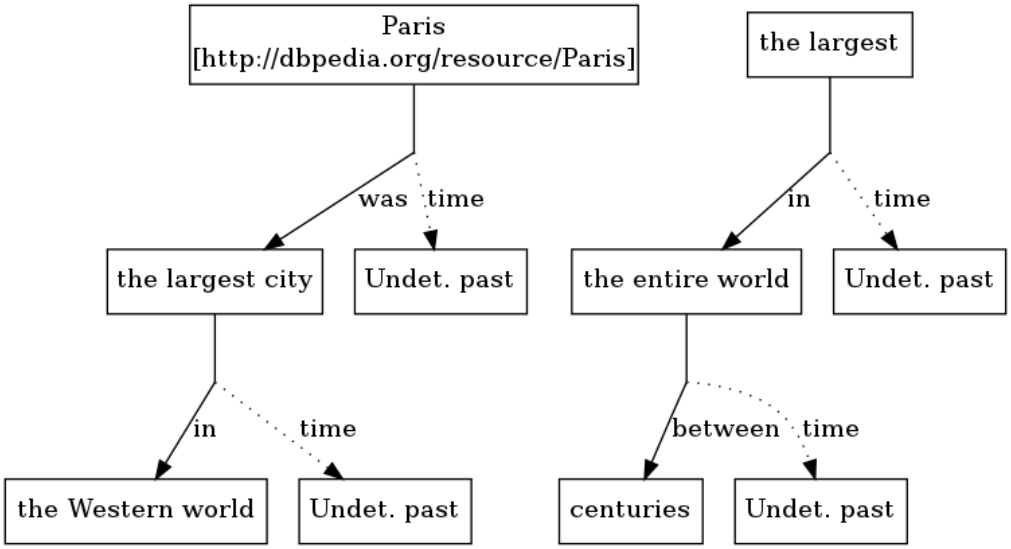


Figura 45: Na sentença *Paris was the largest city in the Western world for about 1,000 years , prior to the 19th century , and the largest in the entire world between the 16th and 19th centuries.* o nó *the largest* ficou desconectado do nó *Paris.*

•

Faltando informação Omissão de termos informativos da sentença no grafo construido. Termos informativos não incluem pontuação, artigos ou conjunções. Exemplos abaixo:

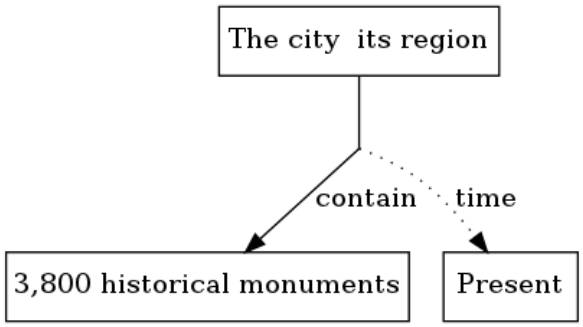


Figura 46: Na Sentença *The city and its region contain 3,800 historical monuments and four UNESCO World Heritage Sites .* a informação após *monuments* foi perdida.

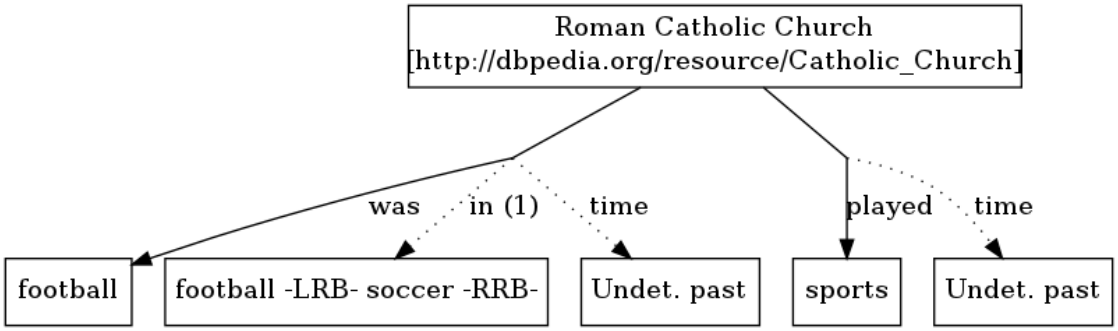


Figura 47: Na sentença *Roman Catholic Church played sports and was particularly interested in football -LRB- soccer -RRB- as a goalkeeper .* o adverbio de modo (*particularly*) e o complemento nominal (*as a goalkeeper* foram perdidos.

-

Atributos da sentença

Os atributos de sentença considerados na avaliação foram:

Contém entidade nomeada Quando a sentença contém uma entidade nomeada. A NE é geralmente um nome conhecido, como "Coca Cola", "Albert Einstein" ou "United States".
Exemplo

Napoleon I of France trained as an officer in mainland France •

Contém referência temporal explícita Quando a sentença contém uma data explícita (seção 5.6.1).
Exemplo

By the early 1990s, Apple was developing alternative platform to the Macintosh, such as the UX. •

Contém correferência pronominal Quando a sentença contém um pronome pessoal ou possessivo.
Exemplo

Instead they sued Microsoft for using a graphical user interface similar to the Apple Lisa.

"They" aqui refere-se à "Apple Inc." •

Contém correferência não-pronominal Quando a sentença contém uma referência a outro termo no texto, e esta referência não é um pronome.
Exemplo

The company was founded by Larry Page and Sergey Brin.

"The company" aqui refere-se à "Google" •

Contém oração subordinada Quando na sentença há uma oração subordinada.
Exemplo

The two theorized about a better system that analyzed the relationships between websites •

Contém coordenação Quando na sentença há pelo menos duas orações coordenadas.
Exemplo

His father was a black foreign student from Kenya and his mother was a white woman from Kansas. •

Sentença mal construída Quando a sentença usa uma forma incomum ou não segue as regras gramaticais da língua. No caso deste trabalho a gramática da língua inglesa.
Exemplo

Paris ' inhabitants are known in English as " Parisians " and in French as " Parisiens ".

"... are known in English as..." é uma construção atípica em texto escrito.
•

Sentença muito complexa Quando a sentença possui construções sintáticas excluídas do escopo do trabalho.

Less than a year later, Napoleon escaped Elba and returned to power, but was defeated at the Battle of Waterloo in June 1815.

A sentença acima é um período com três orações coordenadas. •

Avaliadores e amostras

Dois avaliadores participaram dos experimentos, ambos avaliando o mesmo conjunto de sentenças do conjunto de artigos selecionados: Apple Inc., Google, Napoleon, Paris, Pope John Paul II. Foram avaliadas as primeiras 30 sentenças válidas de cada artigo, ou seja, aquelas em que o pipeline conseguiu gerar um grafo, e que não estivessem na condição de "Sentença muito complexa".

Para cada artigo, foram avaliadas suas duas versões (*English* e *Simple English*), o que resultou em um total de 300 sentenças avaliadas. O número de sentenças avaliadas foi escolhido em função do prazo para conclusão do trabalho, visto que o processo de avaliação consome muito tempo, com algumas sentenças levando mais de dois minutos para serem avaliadas.

O uso de uma quantidade maior de avaliadores implicaria em um grande esforço de padronização dos critérios de avaliação, a fim de conseguir um nível aceitável de concordância (seção 6.2.3), também excedendo o prazo para a conclusão do trabalho.

6.2.2 Sistema de avaliação

Para auxiliar na tarefa de avaliação, foi construído um sistema para o cadastro das informações relativas à avaliação de cada sentença. O sistema foi desenvolvido sobre a *plataforma web*[24] [25], tendo como requisito principal a apresentação de uma interface simples, na qual o avaliador pudesse ver simultaneamente a sentença avaliada e seu grafo produzido, e em seguida marcar os erros e atributos identificados e deixar sua avaliação registrada. Outros requisitos incluiam a identificação dos avaliadores e a possibilidade de acesso rápido às sentenças já avaliadas.

.

Ao utilizá-lo pela primeira vez o avaliador deve registrar um nome e senha, os quais deverão ser utilizados em suas próximas visitas. A interface do sistema é apresentada abaixo:

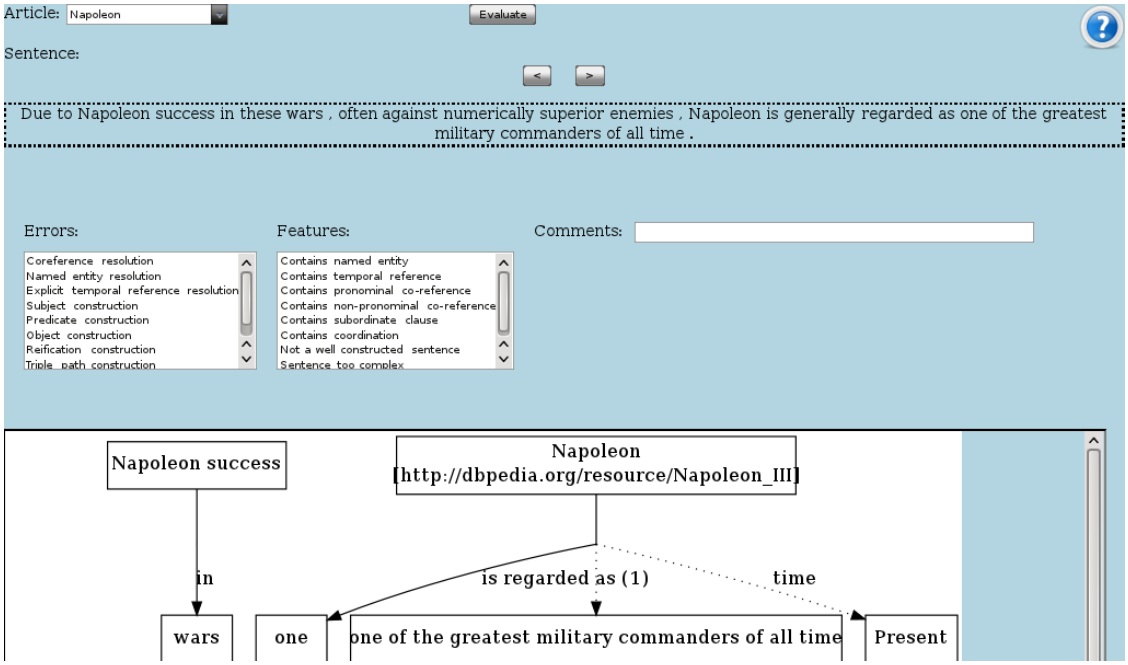
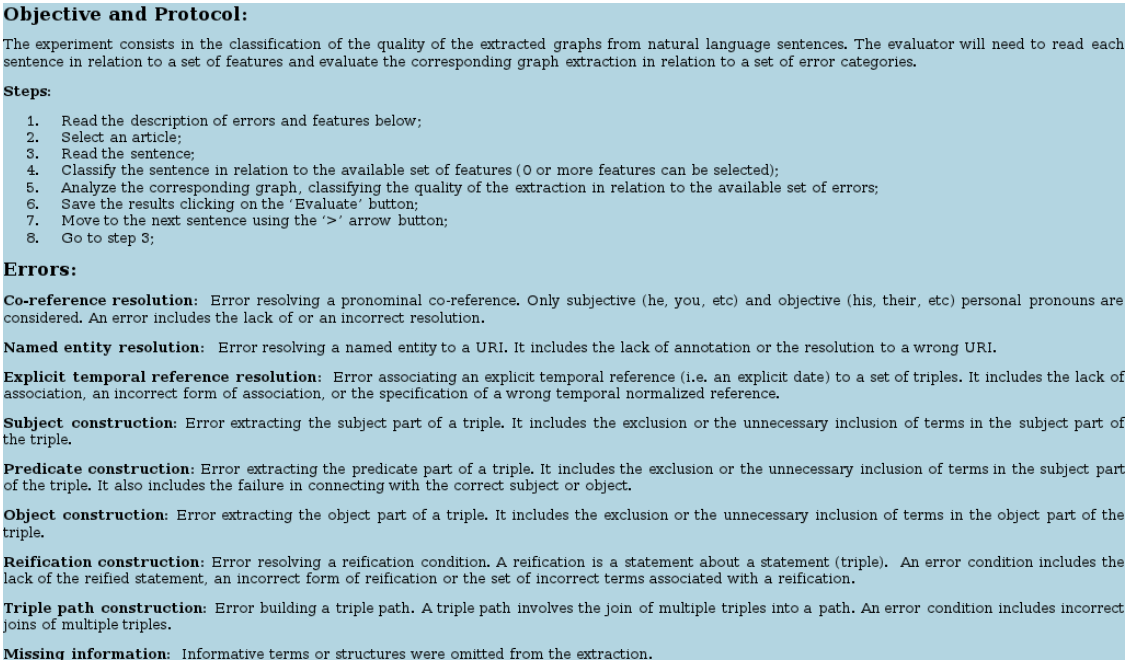


Figura 48: Tela principal do sistema de avaliação.

[24] Páginas HTML em combinação com tecnologias de servidor HTTP, como o Apache (httpd.apache.org).

[25] O sistema de avaliação está disponível em *http://dcc.ufrj.br/danilo/cgi-bin/app.py/Avaliacao*



Objective and Protocol:

The experiment consists in the classification of the quality of the extracted graphs from natural language sentences. The evaluator will need to read each sentence in relation to a set of features and evaluate the corresponding graph extraction in relation to a set of error categories.

Steps:

1. Read the description of errors and features below;
2. Select an article;
3. Read the sentence;
4. Classify the sentence in relation to the available set of features (0 or more features can be selected);
5. Analyze the corresponding graph, classifying the quality of the extraction in relation to the available set of errors;
6. Save the results clicking on the 'Evaluate' button;
7. Move to the next sentence using the '>' arrow button;
8. Go to step 3;

Errors:

Co-reference resolution: Error resolving a pronominal co-reference. Only subjective (he, you, etc) and objective (his, their, etc) personal pronouns are considered. An error includes the lack of or an incorrect resolution.

Named entity resolution: Error resolving a named entity to a URI. It includes the lack of annotation or the resolution to a wrong URI.

Explicit temporal reference resolution: Error associating an explicit temporal reference (i.e. an explicit date) to a set of triples. It includes the lack of association, an incorrect form of association, or the specification of a wrong temporal normalized reference.

Subject construction: Error extracting the subject part of a triple. It includes the exclusion or the unnecessary inclusion of terms in the subject part of the triple.

Predicate construction: Error extracting the predicate part of a triple. It includes the exclusion or the unnecessary inclusion of terms in the subject part of the triple. It also includes the failure in connecting with the correct subject or object.

Object construction: Error extracting the object part of a triple. It includes the exclusion or the unnecessary inclusion of terms in the object part of the triple.

Reification construction: Error resolving a reification condition. A reification is a statement about a statement (triple). An error condition includes the lack of the reified statement, an incorrect form of reification or the set of incorrect terms associated with a reification.

Triple path construction: Error building a triple path. A triple path involves the join of multiple triples into a path. An error condition includes incorrect joins of multiple triples.

Missing information: Informative terms or structures were omitted from the extraction.

Figura 49: Manual de uso do sistema de avaliação.

Os dados das avaliações registradas são armazenados em um arquivo *CSV*[26] para cada avaliador, podendo ser acessados rapidamente através do endereço

http://dcc.ufrj.br/ danilo/arquivos/eval_ rel_ extraction_ <nome>.csv

onde *<nome>* é o nome do avaliador registrado no primeiro acesso. O formato *CSV* tem como caracteristica chave a simplicidade, podendo ser lido com facilidade por seres humanos e pelo computador.

O sistema de avaliação concedeu mais agilidade durante esta etapa do trabalho. Sem este auxílio, não seria possível alcançar um mínimo de avaliações dentro do prazo previsto para o trabalho. Os cálculos foram agilizados pelo formato padronizado das avaliações.

6.2.3 Cálculo dos resultados

Os resultados foram medidos em termos da frequência relativa da ocorrência de cada categoria de erro e de atributo nas sentenças avaliadas. Também foi medido o grau de concordância entre os avaliadores, usando o coeficiente κ *de Cohen* (Cohen, 1960)[12].

O objetivo das medidas foi identificar o estado atual de precisão e a distribuição das falhas ao longo do pipeline, possibilitando o direcionamento de esforços na melhoria dos pontos mais críticos.

[26] *Comma Separated Values*: Formato de representação tabular onde as colunas são separadas por vírgulas.

6.3 Resultados

Dos resultados calculados para o conjunto de sentenças avaliado, foram observadas as seguintes frequências relativas para os atributos encontrados nas sentenças, indicando a fração das sentenças avaliadas que possuia cada atributo:

Atributo	**English**	**Simple English**	**Total**
Nenhum atributo encontrado	1,9%	4,5%	3,1%
Continham entidade nomeada	80,2%	79,5%	79,2%
Continham referência temporal explícita	37,7%	25,0%	29,2%
Continham correferência pronominal	14,1%	26,3%	20,7%
Continham correferência não-pronominal	14,2%	5,8%	10,2%
Continham oração subordinada	18,0%	16,0%	15,6%
Continham coordenação	27,3%	13,0%	18,7%
Sentença mal construída	3,8%	13,5%	8,5%
Sentença muito complexa	68,3%	49,9%	63,8%

Tabela 1: Frequências relativas para os atributos das sentenças

As sentenças classificadas como "muito complexas" não entraram no cálculo das frequências dos erros e da concordância entre os avaliadores.

As frequências relativas para os erros encontrados nos grafos de relações extraídos foram as seguintes:

Erro	**English**	**Simple English**	**Total**
Sem erros encontrados	9,4%	10,9%	14,3%
Resolução de correferência	9,4%	24,4%	16,7%
Resolução de entidade nomeada	56,6%	57,0%	54,4%
Resolução de referência temporal explícita	16,0%	9,6%	11,0%
Construção do sujeito	20,8%	16,0%	17,0%
Construção do predicado	19,8%	9,6%	13,3%
Construção do objeto	27,3%	27,6%	29,6%
Construção de reificação	17,0%	10,1%	13,0%
Construção do caminho de triplas	23,6%	13,5%	16,7%
Faltando informação	24,5%	17,3%	20,0%

Tabela 2: Frequências relativas para os erros em extrações

A concordância entre os avaliadores, calculada por meio do coeficiente κ *de Cohen* para cada categoria de erro foi a seguinte:

Estes resultados serviram para fornecer uma visão mais detalhada dos problemas que estão sendo tratados, indicando as características do texto e permitindo o mapeamento do impacto de cada característica na solução dos problemas.

Mesmo com a propagação de erros intrínseca ao pipeline, as frequências

Erro	English	Simple English	Total
Sem erros encontrados	6,4%	37,8%	30,1%
Resolução de correferência	56,0%	58,8%	58,8%
Resolução de entidade nomeada	24,8%	6,1%	11,4%
Resolução de referência temporal explícita	56,0%	49,5%	53,6%
Construção do sujeito	31,3%	9,7%	23,1%
Construção do predicado	23,2%	4,5%	15,0%
Construção do objeto	0,8%	13,3%	10,3%
Construção de reificação	7,8%	7,6%	7,0%
Construção do caminho de triplas	14,4%	7,0%	5,3%
Faltando informação	16,2%	2,8%	8,3%

Tabela 3: Índices de concordância κ entre os avaliadores para as categorias de erro. (quanto maior melhor)

de ocorrência dos erros caracterizam bem as etapas do pipeline onde os erros ocorrem, permitindo o foco na correção de um problema específico.

Também ficaram claros alguns problemas:

- A frequência obtida para sentenças muito complexas indica que o pipeline em seu estado atual consegue apenas tratar aproximadamente 40% das sentenças comuns ao tipo de discurso usado nos artigos da Wikipedia. Entretanto, este número inclui as sentenças não verbais (aquelas que não possuem verbos), que não fazem parte do escopo do trabalho, sendo portanto um valor subestimado.

- A baixa concordância entre os avaliadores em algumas categorias de erros evidencia uma necessidade de melhoria na especificação da avaliação. É necessária uma definição precisa do que constitui cada erro e o manual de uso ainda não está preciso o suficiente.

- A *Simple English Wikipedia* apresentou um número de sentenças malformadas muito acima do que era esperado. A impressão obtida pelo autor neste ponto é a de que o desejo de simplificar o texto tem levado alguns autores de artigos desta versão a "relaxarem" quanto ao uso da norma culta da língua, adotando construções simples, mas gramaticalmente incorretas ou incomuns.

Com exceção dos casos NER e construção do objeto, os erros ficaram bem distribuidos entre as sentenças analisadas, tendo um impacto menor nas extrações. Por esta razão, os resultados foram considerados positivos, dado o pequeno número de regras usado para alcançá-los (apenas oito), mostrando a viabilidade do tratamento do problema de extração de relações através do uso de regras estruturais. A tolerância a erros é neste caso um fator crucial no processo de melhoria incremental do trabalho, que encontra apoio nos resultados aqui exibidos. A estratégia de melhor esforço é então um pilar fundamental na construção do trabalho, já que permitiu que os resultados das extrações fossem avaliados o mais cedo possível, com isso tornando possível a evolução rápida em todas as etapas do pipeline.

7. Conclusão e trabalhos futuros

Os produtos finais deste trabalho foram:

- Uma interface de comunicação com o *DBpedia Spotlight*: *envia_spotlight*.
- Um normalizador de referências e resolvedor de correferências pronominais: *norm_ne_coref*.
- Um extrator de relações: *extrai_rels_cstruct*.
- Um conjunto básico de regras sintáticas para o extrator.
- Um sorteador/seletor automático de artigos da Wikipedia para avaliação das extrações.
- Um modelo de processamento (pipeline) otimizado para as ferramentas utilizadas.
- Um sistema web para avaliação dos grafos de relações obtidos nas extrações.
- Corpus com o texto limpo de todos os artigos da Wikipedia, tanto na versão "English" quanto na "Simple English".

O corpus obtido está em um formato similar ao utilizado por outros corpora incluidos no NLTK, como o *Reuters-21578 "ApteMod" corpus*, com cada texto diferente separado em um arquivo próprio. Este é um corpus não anotado com grande variedade linguística, pois os textos da Wikipedia tratam de um grande número de assuntos distintos. Atualmente tem um tamanho aproximado de 10GB, e mantém uma tendência de crescimento.

Os resultados obtidos no trabalho apontam a necessidade de melhorias na interface com o *Spotlight* e no extrator de relações, os dois responsáveis pela maior parcela dos erros encontrados.

Uma outra questão a ser tratada é a compatibilização dos dados gerados com a *Linked Data Web*, pois apesar das semelhanças no modelo de dados, ainda há alguns problemas conceituais a serem resolvidos, como a transformação das reificações obtidas para o formato RDF, menos flexível que o adotado neste trabalho.

A melhoria dos resultados de extração e a integração com os padrões da *Linked Data Web* visam aproximar o trabalho de seu objetvo final, que é contribuir com a grande rede aberta de informações que está em construção, visando no futuro possibilitar o acesso rápido e eficiente às informações através de perguntas em linguagem natural.

7.1 Publicações

Este trabalho deu origem à algumas publicações, voltadas à sua inserção no contexto de Extração de Informação e Web Semântica. Em [19] é realizada uma avaliação preliminar do processo de extração, em [17] são descritos os princípios de uma representação desatrelada à ontologias específicas. [20] foca no modelo de representação utilizado (grafos estruturados para o discurso) e [18] apresenta uma demonstração das capacidades do sistema.

Referências Bibliográficas

[1] ISO 8879:1986. Standard generalized markup language (sgml), 1986.

[2] J. C. Azeredo. Iniciação a sintaxe do portugues, 2001.

[3] BBC. Noam chomsky v. ibm's watson computer, http://www.bbc.co.uk/news/technology-12491688, 2011.

[4] E. Bechara. Moderna gramática portuguesa, 2004.

[5] E. Bengtson and D. Roth. Understanding the value of features for coreference resolution. emnlp, 2008.

[6] S. Bird, Klein, E., and E. Loper. Natural language processing with python, 2009.

[7] C. Bizer, S. Auer, G. Kobilarov, J. Lehmann, and R. Cyganiak. Dbpedia – querying wikipedia like a database, 2007.

[8] C. Bizer, T. Heath, and T. Berners-Lee. Linked data — the story so far, 2009.

[9] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. Dbpedia - a crystallization point for the web of data, 2009.

[10] Census. http://www.census.gov/genealogy/names/names_files.html, 2010.

[11] N. Chomsky. Knowledge of language: Its nature, origin, and use, 1986.

[12] J. Cohen. Educat psychol measure; 20: 37-46, 1960.

[13] D. RFC-4627 Crockford. The application/json media type for javascript object notation (json), 2006.

[14] J. R. Curran, S. Clark, and J. Bos. Linguistically motivated large-scale nlp with c&c and boxer, 2007.

[15] A. Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction, 2011.

[16] et al. Ferrucci, D. Building watson: An overview of the deepqa project, 2010.

[17] A Freitas, D. S. Carvalho, and J. C. Pereira da Silva. Extracting linked data graphs from texts: An ontology-agnostic approach. 2013.

[18] A Freitas, D. S. Carvalho, and J. C. Pereira da Silva. Graphia: Extracting contextual relation graphs from text. 2013.

[19] A Freitas, D. S. Carvalho, and S. Pereira da Silva, J. C. O'Riain. A semantic best-effort approach for extracting structured discourse graphs from wikipedia. 2012.

[20] A Freitas, D. S. Carvalho, and S. Curry E. Pereira da Silva, J. C. O'Riain. Representing texts as contextualized entity-centric linked data graphs. 2013.

[21] A. Freitas, J.G. Oliveira, S. O'Riain, E. Curry, and J.C. Pereira da Silva. Querying linked data using semantic relatedness: A vocabulary independent approach, 2011.

[22] GraphViz. http://www.graphviz.org/doc/info/lang.html, 2004.

[23] B. Harrington and S. Clark. Asknet: Creating and evaluating large scale integrated semantic networks, 2008.

[24] E. T. Jaynes. Information theory and statistical mechanics, 1963.

[25] H. Kamp. A theory of truth and semantic representation, 1981.

[26] D. Klein and C. D. Manning. Accurate unlexicalized parsing, 2003.

[27] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data, 2001.

[28] L. Liu and M. T. Özsu. Encyclopedia of database systems, 2009.

[29] C. D. Manning and H. Schütze. Foundations of statistical natural language processing, 1999.

[30] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. Dbpedia spotlight: Shedding light on the web of documents, 2011.

[31] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*, 2011.

[32] N. J. Nilsson. Artificial intelligence: A new synthesis, 1998.

[33] NIST. Message understanding conference (muc), 1987.

[34] P. Peters and A. Smith. The australian corpus of english (ace), 1986.

[35] G. C. Schmitt. Ibm's watson supercomputer crowned jeopardy king, http://www.framingbusiness.net/archives/1287, 2011.

[36] M. Steedman. Combinatory grammars and parasitic gaps, 1987.

[37] V. Stoyanov, C. Cardie, N. Gilbert, E. Riloff, D. Buttler, and D. Hysom. Reconcile: A coreference resolution platform, 2010.

[38] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago - a core of semantic knowledge, 2007.

[39] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A Core of Semantic Knowledge. In *16th international World Wide Web conference (WWW 2007)*, New York, NY, USA, 2007. ACM Press.

[40] Università di Pisa UNIPI/Medialab. http://medialab.di.unipi.it/wiki/wikipedia_extractor, 2009.

[41] University of Pennsynlania UPenn/CIS. http://www.cis.upenn.edu/ treebank, 1995.

[42] W3C. Extensible markup language (xml), 1996.

[43] W3C. Resource description framework (rdf) model and syntax specification, http://www.w3.org/tr/pr-rdf-syntax, 1999.

[44] W3C. Xhtml^{TM} 1.0 the extensible hypertext markup language (second edition), http://www.w3.org/tr/xhtml1, 2000.

[45] W3C. Owl web ontology language reference, http://www.w3.org/tr/owl-ref, 2004.

[46] W3C. Rdfa primer, http://www.w3.org/tr/xhtml-rdfa-primer, 2004.

[47] W3C. Speech synthesis markup language, http://www.w3.org/tr/speech-synthesis/, 2004.

[48] W3C. Sparql query language for rdf, 2008.

[49] P. Wojtinnek, B. Harrington, S. Rudolph, and S. Pulman. Conceptual knowledge acquisition using automatically generated large-scale semantic networks, 2010.