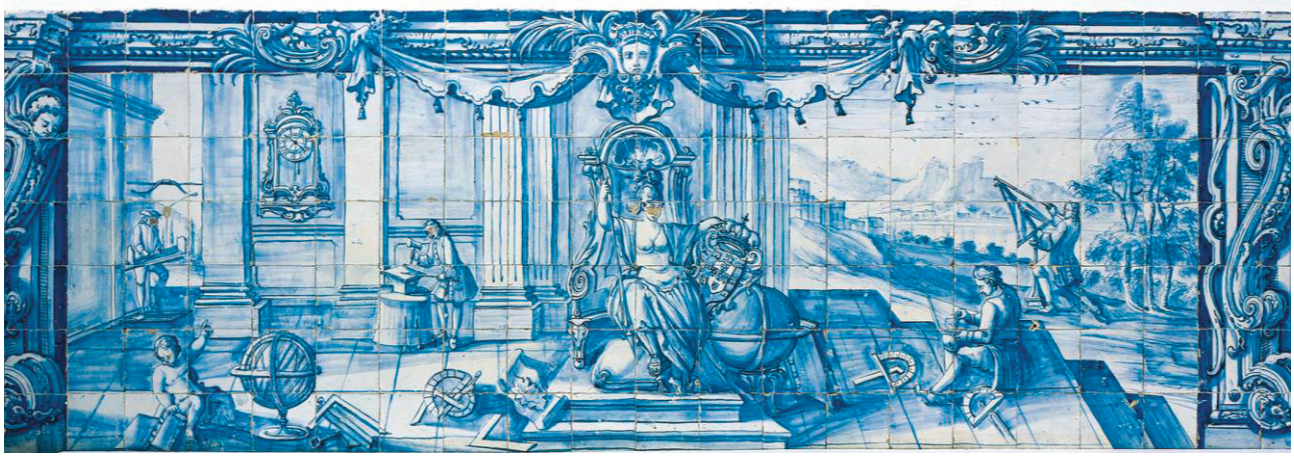
SISTEMAS DE PERGUNTA-RESPOSTA PARA A WEB SEMÂNTICA

Dora Regina Oliveira Melo

Tese apresentada à Universidade de Évora
para obtenção do Grau de Doutor em setembro de 2013
Especialidade: Informática

ORIENTADORES(A/ES): *Irene Pimenta Rodrigues*
Vítor Manuel Beires Pinto Nogueira

ÉVORA, SETEMBRO DE 2013



INSTITUTO DE INVESTIGAÇÃO E FORMAÇÃO AVANÇADA

Doutoramento em Informática

Sistemas de Pergunta-Resposta para a Web Semântica

Dora Regina Oliveira Melo

Orientador
Irene Pimenta Rodrigues

Coorientador
Vítor Manuel Beires Pinto Nogueira

Sumário

O aumento significativo de informação não estruturada e heterogénea, disponível na web, e a necessidade no acesso em tempo real promovem o desenvolvimento de ferramentas, capazes de "dialogar" com o utilizador e que respondam de forma eficaz às suas exigências.

O foco desta tese está no desenvolvimento de um sistema cooperativo de Pergunta-Resposta para a Web Semântica. As principais contribuições consistem no enriquecimento do sistema com um Controlador do Discurso que, através da análise e da representação semântica da questão, do tipo de resposta esperada, e da estrutura do discurso, permite fornecer respostas objetivas, informativas e justificadas. Acrescido de capacidades cooperativas, capacidade de interação com o utilizador, o sistema obtém informação necessária para esclarecer ambiguidades e conduzi-lo no caminho para a resposta correta. O desempenho do sistema é avaliado por comparação com outros sistemas semelhantes. Embora o conjunto de testes seja de pequena dimensão, os resultados obtidos são bastante promissores.

Question Answering for the Semantic Web

Abstract

The significant increase of unstructured and heterogeneous information, available on the web, and the need of real-time access promote the development of engines able to "dialogue" with the user and to answer effectively to their requirements.

The focus of this thesis is in the development of a cooperative Question-Answering system for the Semantic Web. The main contributions consist in enriching the system with a Discourse Controller that, through the analysis and the semantic representation of the question, the type of expected answer and the structure of the discourse, can provide accurate, informative and justified answers. Increased with cooperative skills, ability to interact with the user, the system obtains information necessary to clarify ambiguities and lead on the path to the correct answer. The system performance is evaluated by comparing with similar question answering systems. Although the test suite has slight dimensions, the results obtained are very promising.

Para o meu filho Pedro Nuno, o amor da minha vida

Agradecimentos

Gostaria de expressar, em primeiro lugar, o meu agradecimento especial à Professora Doutora Irene Pimenta Rodrigues e ao Professor Doutor Vítor Manuel Beires Pinto Nogueira, meus orientadores, por todo o apoio, paciência, motivação e, acima de tudo, disponibilidade demonstrados ao longo deste trabalho.

Ao Instituto Superior de Contabilidade e Administração de Coimbra, do Instituto Politécnico de Coimbra, no nome do Excelentíssimo Presidente Dr. Manuel de Sá e Souza de Castelo Branco e Vice-Presidentes Dr. Pedro Miguel Lopes Nunes da Costa e Dr. António Manuel Duarte Gonçalves, o meu mais sincero e especial agradecimento por me terem apoiado na continuidade da minha formação avançada e permitirem a execução do programa inerente à bolsa Protec, possibilitando a dispensa parcial de 50% da minha atividade letiva anual e o pagamento das propinas de doutoramento. Sem este apoio, certamente o trabalho desenvolvido por mim teria sido muito mais difícil.

Ao centro de investigação CENTRIA (Centro de Inteligência Artificial), o meu mais sincero agradecimento, pelo seu fornecimento de fundos que permitiu a minha participação em conferências, tanto nacionais como internacionais, e a publicação de artigos em revistas de âmbito internacional.

À minha família, em especial, ao meu pai e à minha mãe, o meu muito obrigada por todo o apoio incondicional que me deram ao longo destes anos, imprescindível à realização deste trabalho. O meu sincero e especial agradecimento à minha irmã pela amizade, apoio e carinho demonstrado ao longo de todas as nossas vidas.

E por último, mas não menos importante, o meu especial agradecimento ao meu filho Pedro Nuno por existir e por ele todo o meu esforço e trabalho é justificado.

Acrónimos

DCMI	*Dublin Core Metadata Initiative*
DL	*Description Logic*
DRS	*Discourse Representation Structures*
HTML	*HyperText Markup Language*
NLP	*Natural Language Processing*
OWL	*Ontology Web Language*
OWL2	*Ontology Web Language 2*
RDF	*Resoure Description Framework*
RIF	*Rule Interchange Format*
SKOS	*Simple Knowledge Organization System*
SOAP	*Simple Object Access Protocol*
SPARQL	*SPARQL Protocol and RDF Query Language*
TREC	*Text REtrieval Conference*
URI	*Uniform Resource Identifier*
URLs	*Uniform Resource Locators*
W3C	*World Wide Web Consortium*
web	*World Wide Web*
XML	*eXtensible Markup Language*
YAGO	*Yet Another Great Ontology*

Conteúdo

Sumário **i**

Abstract **iii**

Lista de Conteúdo **xiv**

Lista de Figuras **xv**

Lista de Tabelas **xviii**

1 Introdução e Motivação **1**

2 Ontologias e a Web Semântica **5**

2.1 Introdução 5

2.2 Linguagens de Representação de Ontologias 8

2.2.1 Lógicas de Descrição como Linguagens de Representação de Ontologias 9

2.2.2 Linguagem de Representação de Ontologias para a Web Semântica . 9

2.3 Lógicas Descritivas 10

2.3.1 Sintaxe e Semântica duma Linguagem de Descrição 10

2.3.1.1 A Lógica Descritiva básica $\mathcal{AL}$ 11

2.3.1.2 Terminologia 12

2.3.1.3 Descrição do Universo 13

2.3.2 Extensões da Lógica Descritiva Básica 15

2.3.3 Domínios Concretos 19

2.3.3.1 Domínio Concreto Numérico $\mathbb{Q}$ 22

2.3.3.2 Intervalos de Tempo 23

2.3.4 Funções de Agregação 25
2.3.5 Sistemas Computacionais para Lógicas Descritivas 28
2.4 *Resource Description Framework* (RDF) 29
2.5 *Ontology Web Language* (OWL) 32
2.6 SPARQL, uma Linguagem de Consulta para Documentos RDF 36
2.7 DBpedia 38
2.7.1 A Base de Dados DBpedia 39
2.8 WordNet 40
2.9 Conclusão 42

3 Sistemas de Pergunta-Resposta 43
3.1 Introdução 43
3.2 Uma Visão Geral sobre os Sistemas de Pergunta-Resposta 45
3.2.1 Os Sistemas Cooperativos de Pergunta-Resposta 46
3.2.2 Outros Sistemas de Pergunta-Resposta 48
3.3 Arquitetura de um Sistema de Pergunta-Resposta 50
3.4 Metodologias Utilizadas 51
3.5 Desafios de Desenvolver Sistemas de Pergunta-Resposta 52
3.6 Tópicos de Pesquisa Atuais 54
3.7 Processamento da Língua Natural 55
3.8 Estruturas de Representação do Conhecimento 57
3.9 Conclusão 59

4 Ferramenta Cooperativa de PR para OWL 61
4.1 Introdução 61
4.2 Análise Sintática 64
4.3 Interpretação Semântica 65
4.3.1 Classificação das Questões 68
4.4 Contexto Ontológico 69
4.5 Avaliação Semântica 72
4.6 Controlador do Discurso 75
4.7 Conclusão 78

5 Controlador do Discurso 79
5.1 Introdução 79

5.2 Extração da Resposta 81
5.3 Processamento da Resposta 84
5.3.1 Avaliação das Propriedades 85
5.3.1.1 Entropia 86
5.3.1.2 Ganho de Informação 88
5.3.1.3 Razão do Ganho 94
5.3.2 Escolha da Melhor Propriedade 95
5.3.3 Diálogo Controlado com o Utilizador 96
5.4 Conclusão 100

6 Implementação do Sistema Proposto 103
6.1 Introdução 103
6.2 Base de Conhecimento 104
6.2.1 DBpedia 104
6.2.1.1 Acesso à Base de Dados DBpedia 104
6.2.1.2 Ontologia DBpedia 108
6.2.2 WordNet 109
6.3 Análise Sintática e Interpretação Semântica da Questão 110
6.4 Contexto Ontológico 113
6.5 Avaliação Semântica 115
6.6 Controlador do Discurso 118
6.7 Gestor de Diálogos 119
6.8 Processamento da Resposta 123
6.9 Interface do Utilizador 123
6.10 Conclusão 124

7 Avaliação da Aplicação Prática do Sistema 125
7.1 Introdução 125
7.2 Sistemas de PR Utilizados para Comparação 126
7.2.1 START - um Sistema de Pergunta-Resposta Baseado na Web 126
7.2.2 PowerAqua - um sistema de Pergunta-Resposta Baseado em Múltiplas Ontologias para a Web Semântica 128
7.3 Conjunto de Questões Utilizado na Avaliação 129
7.4 Resultados de Comparação - sistemas START e PowerAqua 130
7.5 Resultados de Desempenho 133
7.6 Conclusão 134

8 Conclusões e Trabalho Futuro **135**

Referências Bibliográficas **148**

A Predicados Prolog **151**

A.1 Contexto Ontológico 151

A.2 Avaliação Semântica 154

A.3 Controlador do Discurso 157

Lista de Figuras

2.1 As relações temporais de Allen. 24

2.2 TBox $\mathcal{T}$ com definições básicas de entidades. 27

4.1 Arquitetura do Sistema Cooperativo de Pergunta-Resposta para a Web Semântica. 63

5.1 Arquitetura do Controlador do Discurso. 80

7.1 Exemplo da questão "Who is Anubis?" a ser colocada ao sistema START, utilizando o interface web disponibilizado online. 127

7.2 Resposta devolvida pelo sistema START à questão "Who is Anubis?". . . . 128

7.3 Exemplo da questão "Who is Anubis?" a ser colocada ao sistema PowerAqua-DBpedia, utilizando o interface web disponibilizado online. 130

7.4 Resposta devolvida pelo sistema PowerAqua-DBpedia à questão "Who is Anubis?". 131

Lista de Tabelas

2.1 DL: Compêndio de construtores de conceitos. 19

2.2 DL: Compêndio de construtores de qualidades. 20

2.3 Domínios concretos numéricos e respetivas complexidades. 22

4.1 Descrição de algumas das marcações do discurso *Penn Treebank* para a língua Inglesa. 65

5.1 Alguns recursos da ontologia que correspondem a termos da DRS da questão "Where is the Taj Mahal?". 83

5.2 Recursos da base de conhecimento, entidades das soluções da questão "Where is the Taj Mahal?" que geram ambiguidade. 86

5.3 Cálculo de cada uma das parcelas que compõem a fórmula para determinar o valor da entropia do conjunto T. 88

5.4 Resultados obtidos relativamente à Entropia, Ganho de Informação e Razão do Ganho de Informação das propriedades utilizadas para a clarificação da multiplicidade de soluções da questão "Where is the Taj Mahal?" - primeira iteração da aplicação do Algoritmo 1. 93

5.5 Resultados obtidos relativamente à Entropia, Ganho de Informação e Razão do Ganho de Informação das propriedades utilizadas para a clarificação da multiplicidade de soluções da questão "Where is the Taj Mahal?" - Segunda iteração da aplicação do Algoritmo 1. 99

6.1 Informação sobre atribuição feita às questões de acordo com o seu tipo e resposta esperada. 111

6.2 Resultados obtidos relativamente à Entropia, Ganho de Informação e Razão do Ganho de Informação das propriedades utilizadas para a clarificação da multiplicidade de soluções da questão "Who is Barabara Jordan?" - primeira iteração da aplicação do Algoritmo 1. 121

7.1 Informação dos resultados obtidos no teste de desempenho da nossa proposta e dos sistemas START e PowerAqua-DBpedia. 132

7.2 Informação dos Resultados obtidos na Avaliação do Desempenho do sistema de Pergunta-Resposta para a Web Semântica proposto. 133

Capítulo 1

Introdução e Motivação

A informação é um dos ativos mais valiosos da sociedade moderna. Com a presença generalizada de computadores, o armazenamento de grandes quantidades de informação tornou-se eficiente e de baixo custo. A evolução da Internet permitiu que a informação pudesse ser partilhada e consultada de qualquer parte do mundo. Estamos agora numa posição onde temos uma quantidade sem precedentes de informação de acesso fácil. Como podemos aceder a essas grandes quantidades de informação, de modo a encontrar o que nos interessa?

Quando a informação é armazenada em base de dados, a tarefa de acesso aos dados é mais facilitada. As bases de dados são caracterizadas por conterem informação estruturada, onde esta é identificada através de códigos, parâmetros e palavras-chave, sobre domínios específicos. No entanto, a maioria da informação disponibilizada na web não está disponível em bases de dados estruturadas, mas num formato não estruturado, tal como texto simples, na maioria das vezes heterogénea, cobrindo os mais variados assuntos, sem marcações ou identificações, e muitas das vezes não é proveniente de fontes credíveis, tornando o processo de consulta bastante complexo.

Transformar os dados não estruturados em dados formatados pode ser um processo muito trabalhoso, dependendo da complexidade da base de dados esperada. Como consequência, a maioria dos dados permanecem armazenados num formato não estruturado. Assim, a pesquisa de informação relevante em grandes quantidades de dados não estruturados exige meios automáticos que possam ajudar nesse processo. Nas últimas décadas foram sendo desenvolvidas ferramentas e linguagens de representação que permitem marcar, codificar, representar e raciocinar sobre a informação disponível na web. Este avanço conduziu as comunidades de investigação, na área da Inteligência Artificial, a percorrer novas linhas de pesquisa que permitissem o desenvolvimento de ferramentas inteligentes para a pesquisa

na web.

O surgimento dos motores de busca, sistemas de pesquisa de informação, permitiram através da correspondência entre palavras encontrar os documentos disponíveis na web. A forma de apresentar os resultados através da sua listagem, permite ao utilizador consultar e verificar qual ou quais dos documentos são do seu interesse. No entanto, esta solução, embora eficaz, faz com que os seus utilizadores percam bastante tempo nestas pesquisas e quando não são conhecedores do assunto que pesquisam poderão tornar a consulta uma tarefa difícil, tanto pela dificuldade na escolha da resposta correta, como por não saberem distinguir qual a informação que pode ser credível. O desenvolvimento de métodos e ferramentas de pesquisa de informação relevante de forma automática fazem parte do conjunto de trabalhos desenvolvidos na área de Recuperação da Informação. Estes sistemas permitem que o utilizador através de uma consulta, na forma de uma série de palavras-chave que descrevem as necessidades do utilizador, tenha acesso a uma lista de documentos pertinentes que possam satisfazer as suas necessidades de informação. No entanto, muitas vezes, uma necessidade de informação por parte do utilizador é mais específica e muito mais apropriada se for expressa em língua natural através de uma questão, em vez de um conjunto de palavras-chave. Os sistemas de recuperação de informação que permitem aos utilizadores fazer perguntas em língua natural são conhecidos como sistemas de Pergunta-Resposta.

O desenvolvimento de sistemas que interagem com os utilizadores em língua natural tem sido um dos objetivos da comunidade de investigação na área da Inteligência Artificial, mais concretamente nas comunidades da Recuperação de Informação e Processamento de Língua Natural. Desde 1960, quando o assunto estava no seu início, uma variedade de interfaces em língua natural para a consulta de bases de dados, sistemas de diálogo e sistemas de compreensão da língua, foram criados. No entanto, estes sistemas eram definidos em domínios muito específicos, contendo um conjunto limitado de expressões na língua Inglesa, o que condicionava o acesso eletrónico a um qualquer utilizador. O desenvolvimento de técnicas de Processamento de Língua Natural e o aumento de recursos linguísticos proporcionou que o interesse por interfaces gerais de língua natural aumentasse consideravelmente nas últimas décadas.

O foco desta tese está no desenvolvimento de um sistema de Pergunta-Resposta cooperativo para a Web Semântica. O trabalho a que nos propomos e que é resultado de um trabalho intenso de cerca de 3 anos de pesquisa, caracterização, desenvolvimento e implementação, consiste num sistema de Pergunta-Resposta que recebe questões expressas em língua natural e é capaz de devolver uma resposta cooperativa, i.e., uma resposta objetiva, informativa e justificada, também expressa em língua natural, obtida da base de conhecimento (constituída por ontologias, base de dados online da DBpedia, e recursos lexicais da WordNet). A inovação neste trabalho consiste no enriquecimento do sistema de Pergunta-Resposta com capacidades cooperativas. Por cooperativo consideramos: o processo de interagir com o utilizador de forma a obter informação necessária para esclarecer e conduzir o sistema no caminho certo para a resposta correta; fornecer uma resposta

objetiva, clara, justificada e contendo informação adicional que possa elucidar melhor o utilizador.

A presente tese está articulada em torno de 8 capítulos: 1 - Introdução e Motivação; 2 - Ontologias e a Web Semântica; 3 - Sistemas de Pergunta-Resposta; 4 - Ferramenta Cooperativa de Pergunta-Resposta para OWL; 5 - Controlador do Discurso; 6 - Implementação do Sistema Proposto; 7 - Avaliação da Aplicação Prática do Sistema Proposto; e 8 - Conclusões e Trabalho Futuro.

No capítulo 2 apresentamos os conceitos "ontologias" e "Web Semântica", evidenciando a forma como estão relacionados no contexto da consulta na web. Em adição, apresentamos como estes dois conceitos suportam o desenvolvimento do nosso trabalho através de linguagens de representação, nomeadamente: as Lógicas Descritivas; a família de especificações *Resource Description Framework* (RDF); a linguagem de ontologias padrão para a Web Semântica *Ontology Web Language* (OWL); e a linguagem para interrogar e consultar a Web Semântica *SPARQL Protocol and RDF Query Language*. Finalizamos estabelecendo algumas conclusões finais.

No capítulo 3 apresentamos um resumo sobre os sistemas de Pergunta-Resposta, focando os trabalhos desenvolvidos que estão relacionados com sistemas cooperativos e têm como domínio de conhecimento a Web Semântica. Em adição, apresentamos a arquitetura típica de um sistema de Pergunta-Resposta, introduzimos algumas características sobre metodologias utilizadas mais frequentemente nos sistemas de Pergunta-Resposta, enumeramos um conjunto de desafios inerentes ao desenvolvimento de sistemas de Pergunta-Resposta e apresentamos alguns tópicos de pesquisa atuais. Introduzimos também os conceitos "Estruturas de Representação do Conhecimento" e "Processamento de Língua Natural" necessários para o nosso trabalho. Concluímos com algumas considerações finais.

No capítulo 4 apresentamos a nossa proposta para um sistema Cooperativo de Pergunta-Resposta que recebe questões expressas em língua natural e tem a capacidade de retornar uma resposta cooperativa, também expressa em língua natural, obtida dos recursos na Web Semântica. Começamos por fazer uma introdução do sistema proposto, descrevendo as principais componentes da sua arquitetura, nomeadamente: Interpretação Semântica, Contexto Ontológico, Avaliação Semântica e Controlador do Discurso. Em paralelo, apresentamos um exemplo ilustrativo das funcionalidades do sistema. Para finalizar, apresentamos algumas ideias e conclusões a retirar deste capítulo.

No capítulo 5 apresentamos o Controlador do Discurso, componente principal do sistema proposto. Em adição, estabelecemos o processo de extração das respostas às questões colocadas pelo utilizador, expomos o procedimento efetuado no processamento das respostas por parte do Controlador do Discurso, e finalizamos com a apresentação das conclusões.

No capítulo 6 apresentamos aspetos importantes da implementação do sistema Cooperativo de Pergunta-Resposta apresentado nos capítulos anteriores. Começamos por apresentar a base de conhecimento que constitui a fonte de conhecimento para o trabalho em causa. Depois, apresentamos algumas técnicas e respetivos predicados Prolog desen-

volvidos para implementar os diferentes módulos, que constituem o sistema cooperativo de Pergunta-Resposta para a Web semântica proposto. Para finalizar, estabelecemos as conclusões finais.

No capítulo 7 abordamos e avaliamos os resultados da aplicação prática a que submetemos o sistema apresentado. A implementação do sistema cooperativo de Pergunta-Resposta para a Web Semântica, embora não completa, permitiu a aplicação prática dos procedimentos propostos, cujos resultados podem ser objeto de uma avaliação concreta. Assim, começamos por apresentar a importância da avaliação de desempenho dos sistemas de Pergunta-Resposta, os sistemas de Pergunta-Resposta que utilizamos para comparação, bem como, o conjunto de questões utilizados na avaliação efetuada. Posteriormente expomos os resultados obtidos na comparação do desempenho dos vários sistemas e os resultados de desempenho obtidos pela nossa proposta. Para finalizar, apresentamos as conclusões.

Finalmente, no capítulo 8 apresentamos as conclusões gerais do presente trabalho e apontamos algumas linhas de orientação para a sua continuação.

Capítulo 2

Ontologias e a Web Semântica

Neste capítulo apresentamos os conceitos "ontologias" e "Web Semântica", evidenciando a forma como estão relacionados no contexto da consulta na web. Em adição, apresentamos como estes dois conceitos suportam o desenvolvimento do nosso trabalho através de linguagens de representação. O presente capítulo está estruturado da seguinte forma. Na Secção 2.1 introduzimos os conceitos de ontologia e Web Semântica e a forma como se relacionam, no contexto das consultas na web. Na Secção 2.2, introduzimos o conceito de linguagens de representação das ontologias e nas restantes secções apresentamos as linguagens de representação de ontologias e a sua descrição para a Web Semântica necessárias para o nosso trabalho, nomeadamente: na Secção 2.3, as Lógicas Descritivas; na Secção 2.4, a família de especificações *Resource Description Framework* (RDF); na Secção 2.5, a linguagem de ontologias normalizada para a Web Semântica *Ontology Web Language* (OWL); e na Secção 2.6, a linguagem para interrogar e consultar a Web Semântica *SPARQL Protocol and RDF Query Language*. Nas Secções 2.7 e 2.8 apresentamos, respetivamente, os projetos DBpedia e WordNet, que definem as bases de dados utilizadas no presente trabalho. Finalmente, na Secção 2.9, estabelecemos as conclusões finais.

2.1 Introdução

Uma ontologia é uma especificação explícita de uma conceptualização, esta definição foi originalmente introduzida por Tom Cruber [39] no âmbito da Inteligência Artificial. O termo é emprestado da filosofia, onde uma ontologia é uma exposição (relato) sistemática

da existência. Mais concretamente, e recorrendo ao dicionário online da Porto Editora - Infopédia[1], uma ontologia define-se como sendo uma "parte da metafísica que estuda o ser em si, as suas propriedades e os modos por que se manifesta". Ou seja, podemos entender que uma ontologia é a identificação, a descrição e a definição de conceitos, de entidades, das suas propriedades e características, dos seus comportamentos, bem como das relações que é possível estabelecer entre eles num determinado contexto.

Uma conceptualização pode ser interpretada como sendo uma visão abstrata e simplificada do mundo tal como nós o desejamos representar, destinado a um determinado propósito. Ou seja, uma conceptualização significa um modelo abstrato de alguns aspetos do mundo, que tomam a forma de definições de propriedades de conceitos e de relações importantes. Assim, uma ontologia não é mais do que a descrição (tal como uma especificação formal de um programa) de conceitos e relações que podem existir entre eles, no contexto de um determinado domínio. As ontologias capturam a estrutura intrínseca conceptual do domínio (mundo). Por uma especificação explícita entende-se o modelo expresso numa linguagem não ambígua, que seja percetível e que possa ser processada tanto pelas pessoas como pelas máquinas.

Uma ontologia exprime um vocabulário de representação, frequentemente especializado para representar algum domínio ou assunto, e que constitui o domínio do conhecimento, i.e., a base de conhecimento (campo de atividades onde se compartilha conhecimentos e experiências sobre determinado tema, interesse ou acontecimento). Mais precisamente, não é o vocabulário em si que se qualifica de ontologia, mas a conceptualização que os termos no vocabulário estão destinados a capturar e a representar. Traduzir os termos da ontologia de uma linguagem para outra não altera a conceptualização da ontologia. Assim, um domínio de conhecimento formalmente representado é baseado numa conceptualização: os objetos, conceitos (e outras entidades que se assumem existir, em alguma área de interesse) e as relações que existem entre eles. Qualquer domínio de conhecimento está comprometido com uma conceptualização, explícita ou implicitamente [40].

A definição de ontologia parece gerar grande controvérsia, não existindo uma definição uniforme e concordante no seio da Inteligência Artificial. Ao longo dos últimos 20 anos, foram sendo apresentadas as mais variadas propostas da sua definição, dependendo sempre do contexto da sua aplicação.

Chandrasekaran, Josephson e Benjamins [27] consideram que

> "(...) a análise ontológica clarifica a estrutura do conhecimento. Dado um domínio, a sua ontologia forma o coração de qualquer sistema de representação do conhecimento para esse domínio. Sem ontologias, ou conceptualizações que suportam o conhecimento, não poderá existir um vocabulário para o conhecimento. (...)
>
> As ontologias permitem a partilha do conhecimento."

[1] `http://www.infopedia.pt/lingua-portuguesa/`

Neste contexto, uma ontologia é um conjunto de definições formais de um vocabulário, uma especificação usada para estabelecer compromissos ontológicos. Tom Gruber em [40] define compromisso ontológico como sendo um acordo no sentido de se poder utilizar um vocabulário (efetuar questões e fazer asserções) de uma forma que é consistente (mas não completa) com a teoria especificada pela ontologia.

Atualmente, no âmbito da Inteligência Artificial, uma ontologia é um termo técnico utilizado para denotar um artefacto que foi desenhado para um determinado propósito, que permite a modelação do conhecimento sobre um domínio, real ou imaginário. Para os sistemas de Inteligência Artificial, o que "existe" é o que se pode representar. Quando o domínio do conhecimento é definido através de um formalismo declarativo, o conjunto dos objetos que podem ser representados é denominado de *Universo* do discurso. Este conjunto de objetos, e as relações entre eles, são expressos no vocabulário de representação com o qual um programa baseado em conhecimento representa o conhecimento. Assim, no contexto da Inteligência Artificial, podemos descrever uma ontologia de um programa através da definição dos seus termos de representação. Em tal ontologia, as definições associam nomes de entidades no universo do discurso (i.e., classes, relações, funções, ou outros objetos) com texto percetível pelos humanos e que descreve o que os nomes significam, e axiomas formais que restringem a interpretação e uso correto desses termos. Formalmente, uma ontologia é uma declaração de uma teoria lógica.

A utilização de ontologias, no sentido de unificar os conceitos sobre os mais diversos aspetos de um determinado domínio de conhecimento, permite a troca e reutilização de informação, tanto para sistemas de processamento de informação quanto entre pessoas.

O desenvolvimento de uma camada semântica que trabalha sobre os conteúdos e serviços da web, a Web Semântica, foi apresentada por Tim Berners-Lee [65] e, ao longo da última década, tem vindo a ser reconhecida como o próximo passo na evolução da *World Wide Web* (web). A inclusão de conteúdos semânticos nas páginas web conduz a um conjunto de dados que podem ser direta ou indiretamente trabalhados pelos computadores, permitindo aos utilizadores encontrar, partilhar e combinar esta informação muito mais facilmente.

As ontologias promovem e facilitam a interoperabilidade, o processamento inteligente, a partilha e a reutilização do conhecimento entre os sistemas de informação. Mais recentemente, as ontologias têm sido reconhecidos como um componente importante para a construção da Web Semântica [65]. Este reconhecimento veio introduzir mudanças na forma como as ontologias são construídas e utilizadas. A Web Semântica não é uma web separada, mas uma extensão da atual, na qual a informação tem atribuída um significado bem definido. A Web Semântica é usada para reduzir a ambiguidade da língua natural, permitindo o desenvolvimento de agentes inteligentes. As componentes da Web Semântica são a declaração de significados, a representação do conhecimento, a ontologia (contextualização), os agentes (reage ao ambiente) e a evolução do conhecimento.

As ontologias fornecem a semântica e a representação do conhecimento necessários para a interpretação da Web Semântica. As ontologias colaboram no sentido de se obter uma

web onde os recursos são acessíveis e interpretáveis não só pelos humanos, mas também pelas máquinas através de processos automáticos. As ontologias ajudam as pessoas e os computadores a acederem a informação necessária, a realizarem inferências sobre os objetos do seu domínio, a partilhar informação independentemente do contexto e da linguagem em que estão expressas, i.e., a comunicarem entre si de forma efetiva. Isto é possível tornando explícita a ligação entre a forma e o conteúdo da informação. As ontologias têm um papel crucial no sentido de que permitem o acesso, a interoperabilidade e a comunicação baseados em conteúdo, fornecendo à web um nível de serviço qualitativamente novo.

As ontologias e a Web Semântica [49] tornaram-se em metodologias fundamentais para representar o domínio conceptual do conhecimento, permitindo o aumento das capacidades semânticas de um sistema de Pergunta-Resposta [41]. Estes sistemas por permitirem pesquisas em bases de dados estruturadas, de grande escala, e em bases do conhecimento da Web Semântica podem ser considerados como uma alternativa ou como um complemento aos atuais sistemas de pesquisa na web.

2.2 Linguagens de Representação de Ontologias

No seio da Inteligência Artificial interessa que existam linguagens que, com a sua sintaxe, permitam definir ontologias, bem como a sua partilha e reutilização. De forma a construir uma linguagem de representação do conhecimento baseado na análise, é necessário associar termos com os conceitos e relações existentes na ontologia e desenhar uma sintaxe capaz de representar o conhecimento em termos dos conceitos e relações. Estas linguagens de representação do conhecimento podem ser partilhadas com quem necessitar de representações do conhecimento dentro do mesmo domínio, eliminando, assim, a necessidade de replicação do processo de análise do conhecimento.

O uso efetivo de ontologias requer não só uma linguagem de ontologia bem concebida e bem definida, mas também o apoio de ferramentas de raciocínio. O raciocínio é importante tanto para garantir a qualidade de uma ontologia, como para explorar a estrutura das ontologias e informação baseada em ontologias. O raciocínio pode ser utilizado em diferentes fases do ciclo de vida de ontologias: durante o desenho de ontologias, pode ser utilizado para testar se os conceitos são ou não contraditórios e, também, para obter relações implícitas. Em especial, relativamente a uma ontologia geralmente queremos determinar a hierarquia dos conceitos, i.e., a ordem parcial de conceitos definidos com base na relação de subsunção. A informação de que um conceito é uma especialização de outro, ou que determinados conceitos são sinónimos, pode ser utilizada na fase do desenho da ontologia para testar se as definições dos conceitos na ontologia têm as consequências intencionais ou não.

2.2.1 Lógicas de Descrição como Linguagens de Representação de Ontologias

As ontologias são importantes para muitas aplicações, desde a sua construção, passando pela sua integração e até a sua evolução, dependem muito da disponibilidade de uma semântica bem definida e de ferramentas poderosas de raciocínio. Uma vez que as lógicas descritivas (lógicas de descrição) estão enriquecidas com ambas, são uma das candidatas ideais para linguagens de representação de ontologias.

As lógicas descritivas são uma família de formalismos de representação do conhecimento de um domínio de aplicação (o Universo), em que se define primeiro os conceitos relevantes do domínio (a terminologia) e depois, usando estes conceitos, se especifica propriedades sobre objetos e indivíduos que ocorrem no domínio, ver descrição mais detalhada na Secção 2.3. Esta secção segue uma abordagem semelhante ao trabalho realizado no mestrado em Informática, mas no contexto de problemas de horários, e apresentado em [83]. A adaptação das lógicas descritivas como linguagens de ontologias tem sido destacadas pelo seu importante papel como base na construção de várias linguagens web, incluindo a linguagem *Ontology Web Language* (OWL), e a sua versão mais recente *Ontology Web Language 2* (OWL2), ver Secção 2.5, uma linguagem de ontologias normalizada para a Web Semântica.

2.2.2 Linguagem de Representação de Ontologias para a Web Semântica

Ao longo da última década a web de documentos, tal como é conhecida, foi-se tornando numa web de dados, onde os documentos puderam começar a ser interpretados não apenas pelos humanos, mas também pelos computadores. Os dados representam significados de palavras que estão interligados e, neste âmbito, têm como finalidade atribuir um significado (sentido) aos conteúdos publicados na web de modo que sejam interpretáveis tanto pelos humanos como pelos computadores. O objetivo final da web de dados é permitir que os computadores possam fazer um trabalho mais útil e que seja possível desenvolver sistemas que suportem interações confiáveis através da rede.

Para se chegar a tal web é necessário fornecer à Web Semântica uma linguagem que permita a exportação da web às regras de qualquer sistema de representação do conhecimento. O seu desenvolvimento passa também por adição de diferentes tecnologias aos sistemas informáticos, enriquecê-los com capacidades de entender os documentos disponíveis na web.

A comunidade internacional *World Wide Web Consortium* (W3C)[2] é responsável pelo desenvolvimento de padrões abertos e que permitem assegurar o crescimento a longo prazo da web [50], bem como responsável por construir um conjunto de tecnologias capaz de suportar a web de dados. A W3C define ontologia como sendo "equivalente a uma base

[2] `http://www.w3.org/`

do conhecimento de uma descrição lógica" e a sua visão para o termo "Web Semântica" consiste em ter-se uma web de dados interligados (*linked data*). Os dados interligados definem a essência do conceito "Web Semântica": integração a uma larga escala e de raciocínio sobre os dados web. As tecnologias da Web Semântica permitem que as pessoas possam criar repositórios de dados na web, construir vocabulários, e escrever regras que manipulem esses dados. Os dados interligados são suportados por tecnologias como, entre outras, *Resoure Description Framework* (RDF)[3] (ver Secção 2.4), *SPARQL Protocol and RDF Query Language* (SPARQL)[4] (ver Secção 2.6) e OWL[5] e a sua segunda versão OWL2[6] (ver Secção 2.5).

As ontologias fazem parte do conjunto de padrões desenvolvidos pela W3C para a Web Semântica, são usadas para especificar padrões conceptuais de vocabulários e promover a troca de dados entre sistemas, providenciar serviços para responder a questionários, publicar bases de conhecimento reutilizáveis, e oferecer serviços para facilitar a interoperabilidade entre múltiplos sistemas heterogéneos e bases de dados. O papel principal das ontologias em relação aos sistemas de bases de dados é especificar uma representação de modelação de dados a um nível de abstração acima dos projetos específicos de bases de dados (físico ou lógico), de modo que os dados possam ser exportados, traduzidos, consultados, e unificados através de sistemas e serviços desenvolvidos de forma independente.

2.3 Lógicas Descritivas

2.3.1 Sintaxe e Semântica duma Linguagem de Descrição

As lógicas descritivas são uma família de formalismos de representação do conhecimento e raciocínio sobre ele, constituídas por duas componentes, a TBox e a ABox, [34, 7, 9, 93]. A TBox introduz a *terminologia*, ou seja, o vocabulário do domínio de aplicação (o Universo), enquanto que a ABox contém *asserções* sobre nomes de indivíduos em função do vocabulário, ou seja, é a descrição do Universo.

Uma linguagem de descrição especifica *conceitos*, que correspondem a coleções de indivíduos, e *qualidades* (*roles*), que correspondem a relações binárias entre os indivíduos. Como as relações funcionais são muito frequentes, são distinguidas das restantes relações sendo denominadas por *atributos*.

Dado um conjunto de *conceitos primitivos* (predicados unários) e um conjunto de *qualidades primitivas* (predicados binários), os conceitos complexos são definidos indutivamente usando *construtores* de descrição. As sintaxes das linguagens de descrição diferem umas das outras pelo conjunto de construtores que usam.

[3] `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`
[4] `http://www.w3.org/TR/rdf-sparql-query/`
[5] `http://www.w3.org/TR/owl-ref/`
[6] `http://www.w3.org/TR/owl2-overview/`

Para além dos conjuntos de terminologias e asserções, uma linguagem de descrição também oferece um conjunto de axiomas que relacionam os conceitos e qualidades entre si. Um problema típico para uma terminologia é determinar quando uma descrição é satisfazível, ou quando uma descrição é mais geral que outra, ou seja, quando a primeira subsume a segunda. Um problema importante para a ABox é o de determinar se o seu conjunto de asserções é *consistente*, ou seja, se ele tem um modelo, e se as asserções da ABox garantem que um determinado indivíduo é uma *instância* de um dado conceito descritivo. A *satisfazibilidade* verifica a descrição, e a *consistência* verifica se os conjuntos de asserções são úteis para determinar se uma base de conhecimento faz sentido. Com os testes de *subsunção*, podemos organizar os conceitos da terminologia numa hierarquia de acordo com a sua generalidade.

2.3.1.1 A Lógica Descritiva básica $\mathcal{AL}$

Sejam `A`, `B` conceitos primitivos, `p` uma qualidade primitiva e `C`, `D` conceitos. Na linguagem de descrição básica $\mathcal{AL}$ os conceitos são definidos pelas seguintes regras:

`B` : −	`A`	(conceito primitivo)
	⊤	(conceito universal)
	⊥	(conceito vazio)
	¬ `A`	(negação primitiva)
	`C` ⊓ `D`	(intersecção)
	∀ `p`.`C`	(restrição de valor)
	∃ `p`.⊤	(quantificador existencial limitado)

Exemplo 2.3.1 Suponhamos que `Mausoleum`, `Monument` e `City` são conceitos primitivos. Então

$$\texttt{Monument} \sqcap \texttt{Mausoleum}$$

$$\texttt{Monument} \sqcap [\neg\, \texttt{Mausoleum}]$$

são conceitos $\mathcal{AL}$ e descrevem os monumentos que são mausoléus, e os monumentos que não são mausoléus. Supondo que `located_in` é uma qualidade primitiva, podemos formar os conceitos

$$\texttt{Mausoleum} \sqcap [\exists\, \texttt{located\_in}.\top]$$

e

$$\texttt{Mausoleum} \sqcap [\forall\, \texttt{located\_in}.\texttt{City}]$$

que traduzem as frases "os mausoléus localizados em algum lugar", e "os mausoléus que estão localizados apenas em cidades", respetivamente. Usando o conceito vazio, podemos construir o conceito

$$\texttt{Mausoleum} \sqcap [\exists\, \texttt{located\_in}.\bot].$$

que descreve "os mausoléus sem localização". ▲

Uma interpretação $\mathcal{I}$ é um par $(\Delta^{\mathcal{I}}, .^{\mathcal{I}})$, onde $\Delta^{\mathcal{I}}$ é um conjunto não vazio denominado domínio da interpretação e $.^{\mathcal{I}}$ é a função de interpretação. A função de interpretação atribui a cada conceito primitivo $\mathtt{A}$ um conjunto $\mathtt{A}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ e a cada qualidade primitiva $\mathtt{p}$ uma relação $\mathtt{p}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. A função de interpretação é estendida à descrição de conceitos através das seguintes definições indutivas:

$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} &= \emptyset \\
(\neg\, \mathtt{A})^{\mathcal{I}} &= \Delta^{\mathcal{I}} \backslash \mathtt{A}^{\mathcal{I}} = \overline{\mathtt{A}^{\mathcal{I}}} = \mathcal{C}\mathtt{A}^{\mathcal{I}} \\
(\mathtt{C} \sqcap \mathtt{D})^{\mathcal{I}} &= \mathtt{C}^{\mathcal{I}} \cap \mathtt{D}^{\mathcal{I}} \\
(\forall\, \mathtt{p}.\mathtt{C})^{\mathcal{I}} &= \{\mathtt{a} \in \Delta^{\mathcal{I}} | \forall \mathtt{b} \in \Delta^{\mathcal{I}}, (\mathtt{a}, \mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \Rightarrow \mathtt{b} \in \mathtt{C}^{\mathcal{I}}\} \\
(\exists\, \mathtt{p}.\top)^{\mathcal{I}} &= \{\mathtt{a} \in \Delta^{\mathcal{I}} | \exists \mathtt{b} \in \Delta^{\mathcal{I}}, (\mathtt{a}, \mathtt{b}) \in \mathtt{p}^{\mathcal{I}}\}
\end{aligned}$$

2.3.1.2 Terminologia

Numa linguagem de descrição a terminologia, TBox, é constituída por um conjunto de definições de conceitos. Ou seja, é o domínio da aplicação constituída pela especificação dos conceitos e propriedades relevantes do domínio. A forma básica de declaração na TBox é a definição de conceitos e/ou qualidades, ou seja, é a criação de novos conceitos a partir de outros já definidos. As definições são utilizadas para introduzir *nomes* de descrições complexas. Na TBox, as definições são especificadas através de *axiomas da terminologia.*

Definição 2.3.1 *Um axioma da terminologia é uma regra que estabelece como os conceitos e as qualidades estão relacionados entre si, utilizando os construtores da linguagem.*

Os axiomas da terminologia podem ter a forma

$$\begin{aligned}
C \equiv D \quad &\text{ou} \quad p \equiv q \\
C \sqsubseteq D \quad &\text{ou} \quad p \sqsubseteq q
\end{aligned}$$

onde $\mathtt{C}$, $\mathtt{D}$ são conceitos e $\mathtt{p}$, $\mathtt{q}$ são qualidades. Os axiomas do primeiro tipo são *igualdades* e os do segundo tipo *inclusões* (hierarquia).

Definição 2.3.2 *Uma* definição *é uma igualdade cujo lado esquerdo é um conceito primitivo.*

Exemplo 2.3.2 Consideremos `Mausoleum` e `City` dois conceitos primitivos, e `located_in` uma qualidade primitiva. O axioma

$$\mathtt{Mausoleum\_at\_City} \equiv [\mathtt{Mausoleum} \sqcap [\forall\, \mathtt{located\_in}.\mathtt{City}]]$$

estabelece a atribuição do nome `Mausoleum_at_City` à descrição do lado direito. Ou seja, o nome `Mausoleum_at_City` representa os mausoléus localizados apenas em cidades. ▲

A semântica dos axiomas é definida da seguinte forma:

Propriedade 2.3.1 *Uma interpretação $\mathcal{I}$ satisfaz uma igualdade $C \equiv D$ se $\mathtt{C}^{\mathcal{I}} = \mathtt{D}^{\mathcal{I}}$, e satisfaz uma inclusão $C \sqsubseteq D$ se $\mathtt{C}^{\mathcal{I}} \subseteq \mathtt{D}^{\mathcal{I}}$. No primeiro caso, diz-se que os conceitos* `C` *e* `D` *são equivalentes. No segundo caso, diz-se que o conceito* `D` *subsume o conceito* `C`*.*

Propriedade 2.3.2 *Se $\mathcal{T}$ é um conjunto de axiomas, então a interpretação $\mathcal{I}$ satisfaz $\mathcal{T}$ se e só se $\mathcal{I}$ satisfaz cada um dos elementos de $\mathcal{T}$.*

Definição 2.3.3 *Se a interpretação $\mathcal{I}$ satisfaz um axioma (respetivamente um conjunto de axiomas), então diz-se que é um modelo daquele axioma (respetivamente do conjunto de axiomas). Dois axiomas ou dois conjuntos de axiomas são equivalentes se são satisfeitos pelas mesmas interpretações, ou seja, se os respetivos modelos são iguais.*

No entanto, existem duas regras importantes que normalmente são desejáveis na construção da terminologia:

- apenas permitir uma definição para o nome de um conceito;
- as definições serem acíclicas, ou seja, os conceitos não serem definidos recursivamente nem em função de outros conceitos que estão relacionados indiretamente com eles.

Estas regras permitem que todos os conceitos definidos se possam expandir de uma forma única numa expressão complexa, contendo apenas conceitos primitivos, por substituição de todos os conceitos definidos.

2.3.1.3 Descrição do Universo

A segunda componente de uma linguagem de descrição é a ABox, descrição do Universo. Na Abox descreve-se um conjunto específico de asserções de qualidades de um domínio de aplicação em função dos conceitos e qualidades. Alguns dos conceitos e qualidades primitivos na ABox podem ser nomes definidos na TBox. Na descrição do Universo são definidos os indivíduos, dando-lhes nomes e estabelecendo propriedades sobre eles, ou seja, são estabelecidas *asserções* sobre os indivíduos em função dos conceitos.

Consideremos `a`, `b`, `c` nomes de indivíduos, `C` um conceito e `p` uma qualidade, as asserções possíveis na ABox são dos seguintes dois tipos:

$$\mathtt{C(a)}, \quad \mathtt{p(b,c)}$$

As asserções do primeiro tipo são denominadas por *asserções de conceitos* - significa que a pertence à interpretação de C, e as do segundo tipo são denominadas de *asserções de qualidades* - significa que b e c estão na relação associada a p.

Definição 2.3.4 *Seja $\mathcal{I}$ uma interpretação. O indivíduo $\mathtt{m} \in \Delta^{\mathcal{I}}$ chama-se instância do conceito $\mathtt{C}$ se $\mathtt{m} \in \mathtt{C}^{\mathcal{I}}$. Se o elemento $(\mathtt{m}_1, \mathtt{m}_2) \in \mathtt{p}^{\mathcal{I}}$ (p uma qualidade), então o indivíduo $\mathtt{m}_2$ é denominado de p-sucessor do indivíduo $\mathtt{m}_1$.*

Exemplo 2.3.3 Se TAJMAHAL e INDIA são nomes de indivíduos, então

$$\mathtt{Mausoleum}(\mathtt{TAJMAHAL})$$

significa que Taj Mahal é um mausoléu, onde Mausoleum é um conceito, e

$$\mathtt{located\_in}(\mathtt{TAJMAHAL}, \mathtt{INDIA})$$

significa que a localização de Taj Mahal é na Índia, onde located_in é uma qualidade. Uma ABox é um conjunto finito de tais asserções. ▲

Uma descrição do Universo pode ser vista como uma instância de uma base de dados relacional contendo apenas relações unárias (conceitos) e relações binárias (qualidades).

A semântica da ABox é dada por expansão da interpretação a nomes de indivíduos. Assim, uma interpretação $\mathcal{I}$ não só faz a correspondência entre conceitos e qualidades, mas também faz a correspondência entre cada nome individual a e um elemento $\mathtt{a}^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Esta correspondência é unívoca, ou seja:

Propriedade 2.3.3 *Se a e b são nomes de indivíduos distintos então $\mathtt{a}^{\mathcal{I}} \neq \mathtt{b}^{\mathcal{I}}$, para a interpretação $\mathcal{I}$.*

Definição 2.3.5 *A interpretação $\mathcal{I}$ satisfaz a asserção do conceito $\mathtt{C}(\mathtt{a})$ se $\mathtt{a}^{\mathcal{I}} \in \mathtt{C}^{\mathcal{I}}$, e satisfaz a asserção da qualidade $\mathtt{p}(\mathtt{a}, \mathtt{b})$ se $(\mathtt{a}^{\mathcal{I}}, \mathtt{b}^{\mathcal{I}}) \in \mathtt{p}^{\mathcal{I}}$.*

Uma interpretação $\mathcal{I}$ satisfaz uma ABox se satisfaz cada asserção da descrição do domínio dessa ABox. Neste caso, dizemos que $\mathcal{I}$ é um modelo das asserções ou da ABox. No entanto, $\mathcal{I}$ satisfaz uma asserção α, ou uma ABox com respeito à TBox, se para além de ser um modelo de α ou da ABox, é um modelo da TBox. Um modelo da ABox e da TBox é uma abstração do universo concreto, onde os conceitos são interpretados como subconjuntos do domínio, como é requerido pela TBox, e onde a afetação dos indivíduos aos conceitos e qualidades respeitam as asserções da ABox.

2.3.2 Extensões da Lógica Descritiva Básica

Se à linguagem de descrição básica $\mathcal{AL}$ adicionarmos mais construtores é possível obtermos uma maior expressividade na linguagem [22, 23, 21, 103, 8]. Cada novo construtor é identificado com uma letra e se adicionarmos a $\mathcal{AL}$ algum dos subconjuntos dos construtores forma-se uma linguagem particular. Designamos cada uma dessas linguagens por

$$\mathcal{AL}\,[\mathcal{U}]\,[\mathcal{E}]\,[\mathcal{N}]\,[\mathcal{Q}]\,[\mathcal{C}]\,[\mathcal{F}]\,[\mathcal{O}]\,[I]\,[\mathcal{H}]$$

onde cada letra está associada ao correspondente construtor adicionado. Vamos expor de seguida alguns desses construtores.

- A *união* de conceitos (indicada pela letra $\mathcal{U}$) é representada por $\mathtt{C} \sqcup \mathtt{D}$, e interpretada como
$$(\mathtt{C} \sqcup \mathtt{D})^{\mathcal{I}} = \mathtt{C}^{\mathcal{I}} \cup \mathtt{D}^{\mathcal{I}}.$$

- O *quantificador existencial* (indicada pela letra $\mathcal{E}$) é representado por $\exists\, \mathtt{p}.\mathtt{C}$, e interpretado como:
$$(\exists\, \mathtt{p}.\mathtt{C})^{\mathcal{I}} = \{\mathtt{a} \in \Delta^{\mathcal{I}} | \exists \mathtt{b} \in \Delta^{\mathcal{I}} : (\mathtt{a},\mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \ \wedge \ \mathtt{b} \in \mathtt{C}^{\mathcal{I}}\}.$$
De notar que $\exists\, \mathtt{p}.\mathtt{C}$ difere de $\exists\, \mathtt{p}.\top$ no facto de permitir que conceitos arbitrários possam ocorrer no espaço do construtor quantificador existencial.

- As *restrições numéricas* (indicadas pela letra $\mathcal{N}$) são $\mathbf{atleast}(\mathtt{n},\mathtt{p})$ e $\mathbf{atmost}(\mathtt{n},\mathtt{p})$, onde $\mathtt{n}$ é um inteiro não negativo e $\mathtt{p}$ é uma qualidade, são interpretadas como:
$$(\mathbf{atleast}(\mathtt{n},\mathtt{p}))^{\mathcal{I}} = \{\mathtt{a} \in \Delta^{\mathcal{I}} | \#(\{\mathtt{b} | (\mathtt{a},\mathtt{b}) \in \mathtt{p}^{\mathcal{I}}\}) \geq \mathtt{n}\}$$
e
$$(\mathbf{atmost}(\mathtt{n},\mathtt{p}))^{\mathcal{I}} = \{\mathtt{a} \in \Delta^{\mathcal{I}} | \#(\{\mathtt{b} | (\mathtt{a},\mathtt{b}) \in \mathtt{p}^{\mathcal{I}}\}) \leq \mathtt{n}\}$$
respetivamente.

- As *restrições numéricas quantificáveis* (indicadas pela letra $\mathcal{Q}$) são $\mathbf{atleast}(\mathtt{n},\mathtt{p}\ \mathbf{is}\ \mathtt{C})$ e $\mathbf{atmost}(\mathtt{n},\mathtt{p}\ \mathbf{is}\ \mathtt{C})$, onde $\mathtt{n}$ é um inteiro não negativo, $\mathtt{p}$ é uma qualidade e $\mathtt{C}$ é um conceito, são interpretadas, respetivamente, por
$$(\mathbf{atleast}(\mathtt{n},\mathtt{p}\ \mathbf{is}\ \mathtt{C}))^{\mathcal{I}} = \{\mathtt{a} \in \Delta^{\mathcal{I}} | \#(\{\mathtt{b} | (\mathtt{a},\mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \wedge \mathtt{b} \in \mathtt{C}^{\mathcal{I}}\}) \geq \mathtt{n}\}$$
e
$$(\mathbf{atmost}(\mathtt{n},\mathtt{p}\ \mathbf{is}\ \mathtt{C}))^{\mathcal{I}} = \{\mathtt{a} \in \Delta^{\mathcal{I}} | \#(\{\mathtt{b} | (\mathtt{a},\mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \wedge \mathtt{b} \in \mathtt{C}^{\mathcal{I}}\}) \leq \mathtt{n}\}.$$

- A *negação* de conceitos arbitrários (indicada pela letra $\mathcal{C}$, de complemento) é escrita como $\neg\, \mathtt{C}$, e interpretada da seguinte forma:
$$(\neg\, \mathtt{C})^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus \mathtt{C}^{\mathcal{I}}.$$

Exemplo 2.3.4 Com este conjunto de construtores adicionais podemos descrever novos conceitos, tais como os mausoléus construídos há pelo menos 500 anos:

$$\texttt{Mausoleum} \sqcap \textbf{atleast}(500, \texttt{built}),$$

onde `Mausoleum` é um conceito primitivo e `built` é uma qualidade primitiva. ▲

- O construtor **sameas** (indicado pela letra $\mathcal{F}$), define todos os elementos que estão em simultâneo em duas qualidades. Ou seja,

$$\textbf{sameas}(\texttt{p}, \texttt{q}) = \{\texttt{a} \in \Delta^{\mathcal{I}} | \exists \texttt{b} \in \Delta^{\mathcal{I}} : \texttt{b} = \texttt{p}^{\mathcal{I}}(\texttt{a}) = \texttt{q}^{\mathcal{I}}(\texttt{a})\}.$$

Algumas vezes é conveniente permitir nomes individuais (também chamados de nominais) não apenas na ABox mas também na linguagem de descrição. Os nomes individuais na definição de conceitos são interpretados como conjuntos singulares, que consistem apenas num elemento do domínio.

- O mais básico dos construtores de nominais é escrito como $\textbf{one\_of}(\texttt{b}_1, ..., \texttt{b}_n)$, onde $\texttt{b}_1, ..., \texttt{b}_n$ são nomes de indivíduos (identificado com a letra $\mathcal{O}$), e é interpretado como

$$(\textbf{one\_of}(\texttt{b}_1, ..., \texttt{b}_n))^{\mathcal{I}} = \{\texttt{b}_1{}^{\mathcal{I}}, ..., \texttt{b}_n^{\mathcal{I}}\}.$$

- Outro construtor envolvendo nominais é o construtor **is**, escrito como `p` **is** `b`, para uma qualidade `p`. A semântica deste construtor é definida como

$$(\texttt{p} \textbf{ is } \texttt{b})^{\mathcal{I}} = \{\texttt{d} \in \Delta^{\mathcal{I}} | (\texttt{d}, \texttt{b}^{\mathcal{I}}) \in \texttt{p}^{\mathcal{I}}\}.$$

  Ou seja, `p` **is** `b` é o conjunto de objetos que conjuntamente com o nominal `b` estão na relação `p`.

- O construtor de qualidades *produto de conceitos* é escrito como $\textbf{product}(\texttt{C}, \texttt{D})$ e interpretado como

$$(\textbf{product}(\texttt{C}, \texttt{D}))^{\mathcal{I}} = \texttt{C}^{\mathcal{I}} \times \texttt{D}^{\mathcal{I}} = \{(\texttt{a}, \texttt{b}) | \texttt{a} \in \texttt{C}^{\mathcal{I}} \wedge \texttt{b} \in \texttt{D}^{\mathcal{I}}\}$$

Para além de todos os construtores de conceitos e qualidades descritos, o seguinte conjunto de construtores de qualidade enriquece bastante a linguagem de descrição permitindo a elaboração de conceitos mais complexos.

- A qualidade *identidade* é identificada como **Id** (indicado pela letra $\mathcal{I}$), corresponde à identidade e é interpretada como

$$(\textbf{Id})^{\mathcal{I}} = \{(\texttt{a}, \texttt{a}) | \texttt{a} \in \Delta^{\mathcal{I}}\}$$

- A *intersecção entre qualidades* é escrita como $\mathtt{p} \sqcap \mathtt{q}$ e interpretada da seguinte forma:
$$(\mathtt{p} \sqcap \mathtt{q})^{\mathcal{I}} = \mathtt{p}^{\mathcal{I}} \cap \mathtt{q}^{\mathcal{I}}$$

- À *união de qualidades* corresponde a qualidade $\mathtt{p} \sqcup \mathtt{q}$ e é interpretada como
$$(\mathtt{p} \sqcup \mathtt{q})^{\mathcal{I}} = \mathtt{p}^{\mathcal{I}} \cup \mathtt{q}^{\mathcal{I}}$$

- O *complementar da qualidade* $\mathtt{p}$ é a qualidade $\neg\, \mathtt{p}$ e é interpretada como
$$(\neg\, \mathtt{p})^{\mathcal{I}} = (\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus \mathtt{p}^{\mathcal{I}}$$

- O construtor de qualidades *hierarquia de qualidades* (indicado pela letra $\mathcal{H}$), identificado como **sub**, define a relação de inclusão entre duas qualidades e é interpretado como
$$\mathbf{sub}(\mathtt{p}, \mathtt{q}) = \mathtt{p}^{\mathcal{I}} \subseteq \mathtt{q}^{\mathcal{I}} = \{(\mathtt{a}, \mathtt{b}) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} | (\mathtt{a}, \mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \Rightarrow (\mathtt{a}, \mathtt{b}) \in \mathtt{q}^{\mathcal{I}}\}$$

- A *relação inversa* de uma qualidade $\mathtt{p}$ é escrita como **inverse** $\mathtt{p}$ (indicada pela letra I) e interpretada da seguinte forma
$$(\mathbf{inverse}\ \mathtt{p})^{\mathcal{I}} = \{(\mathtt{a}, \mathtt{b}) | (\mathtt{b}, \mathtt{a}) \in \mathtt{p}^{\mathcal{I}}\}$$

- O construtor de qualidades **restrict** constrói para uma dada qualidade $\mathtt{p}$ e um conceito $\mathtt{C}$ o conjunto dos elementos $\mathtt{p}(\mathtt{a}, \mathtt{b})$, para os quais $\mathtt{b}$ está no conceito $\mathtt{C}$. É escrito da forma $\mathbf{restrict}(\mathtt{p}, \mathtt{C})$, e interpretado como
$$(\mathbf{restrict}(\mathtt{p}, \mathtt{C}))^{\mathcal{I}} = \{(\mathtt{a}, \mathtt{b}) \in \mathtt{p}^{\mathcal{I}} | \mathtt{b} \in \mathtt{C}^{\mathcal{I}}\}.$$

- A *composição de qualidades* é escrita como $\mathtt{p} \cdot \mathtt{q}$ e interpretada da seguinte forma:
$$(\mathtt{p} \cdot \mathtt{q})^{\mathcal{I}} = \mathtt{p}^{\mathcal{I}} \circ \mathtt{q}^{\mathcal{I}} = \{(\mathtt{a}, \mathtt{c}) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} | \exists \mathtt{b} \in \Delta^{\mathcal{I}} : (\mathtt{a}, \mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \wedge (\mathtt{b}, \mathtt{c}) \in \mathtt{q}^{\mathcal{I}}\}.$$

Exemplo 2.3.5 Consideremos os conceitos primitivos `City` e `Mausoleum`, e a qualidade primitiva `located_in`. É possível, por exemplo, exprimirmos as cidades onde estão localizados mausoléus
$$\mathtt{City} \sqcap [\forall\, [\mathbf{inverse}\ \mathtt{located\_in}].\mathtt{Mausoleum}].$$
Ou então, apenas a localização do Taj Mahal
$$\forall\, [\mathbf{inverse}\ \mathtt{located\_in}].\mathtt{TAJMAHAL},$$
onde `TAJMAHAL` é o nome de um indivíduo. ▲

Os construtores de qualidades conferem às linguagens a possibilidade de expressarmos uma grande quantidade de conceitos complexos.

Exemplo 2.3.6 Suponhamos que, dentro do contexto de uma escola, pretendemos saber quais as disciplinas que são lecionadas por algum professor

$$\texttt{Disciplina} \sqcap [\forall\, [\textbf{inverse}\ \texttt{lecciona}].\texttt{Professor}],$$

onde `Evento` e `Professor` são conceitos primitivos, e `lecciona` é uma qualidade primitiva. Ou ainda, pretendemos saber todas as disciplinas que ocorrem em horários da preferência dos professores que os lecionam:

$$\texttt{Disciplina} \sqcap [\forall\, [[\texttt{ocorre} \cdot \texttt{pref}] \sqcap [\textbf{inverse}\ \texttt{lecciona}]].\texttt{Professor}], \tag{2.1}$$

onde `ocorre` e `pref` são qualidades primitivas.
Consideremos $\texttt{b}.$ como abreviatura de $\texttt{b} \in \Delta^{\mathcal{I}}$. A interpretação semântica do conceito (2.1) é estabelecida da seguinte forma:

$$\begin{aligned}
&(\texttt{Disciplina} \sqcap [\forall\, [[\texttt{ocorre} \cdot \texttt{pref}] \sqcap [\textbf{inverse}\ \texttt{lecciona}]].\texttt{Professor}])^{\mathcal{I}} = \\
&= \texttt{Disciplina}^{\mathcal{I}} \cap (\forall\, [[\texttt{ocorre} \cdot \texttt{pref}] \sqcap [\textbf{inverse}\ \texttt{lecciona}]].\texttt{Professor})^{\mathcal{I}} = \\
&= \{\texttt{a} \in \Delta^{\mathcal{I}} | \texttt{a} \in \texttt{Disciplina}^{\mathcal{I}}\} \cap \{\texttt{a} \in \Delta^{\mathcal{I}} | \forall \texttt{b}.(\texttt{a},\texttt{b}) \in ([\texttt{ocorre} \cdot \texttt{pref}] \sqcap \\
&\quad [\textbf{inverse}\ \texttt{lecciona}])^{\mathcal{I}} \Rightarrow \texttt{b} \in \texttt{Professor}^{\mathcal{I}}\} = \\
&= \{\texttt{a} \in \Delta^{\mathcal{I}} | \texttt{a} \in \texttt{Disciplina}^{\mathcal{I}} \wedge \forall \texttt{b}.(\texttt{a},\texttt{b}) \in ([\texttt{ocorre} \cdot \texttt{pref}] \sqcap [\textbf{inverse}\ \texttt{lecciona}])^{\mathcal{I}} \\
&\quad \Rightarrow \texttt{b} \in \texttt{Professor}^{\mathcal{I}}\} = \\
&= \{\texttt{a} \in \texttt{Disciplina}^{\mathcal{I}} | \forall \texttt{b}.(\texttt{a},\texttt{b}) \in ([\texttt{ocorre} \cdot \texttt{pref}] \sqcap [\textbf{inverse}\ \texttt{lecciona}])^{\mathcal{I}} \\
&\quad \Rightarrow \texttt{b} \in \texttt{Professor}^{\mathcal{I}}\}
\end{aligned}$$

Como a semântica do conceito $[\texttt{ocorre} \cdot \texttt{pref}] \sqcap [\textbf{inverse}\ \texttt{lecciona}]$ é

$$\begin{aligned}
([\texttt{ocorre} \cdot \texttt{pref}] \sqcap [\textbf{inverse}\ \texttt{lecciona}])^{\mathcal{I}} &= \\
&= (\texttt{ocorre} \cdot \texttt{pref})^{\mathcal{I}} \cap (\textbf{inverse}\ \texttt{lecciona})^{\mathcal{I}} = \\
&= \{(\texttt{a},\texttt{b}) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} | \exists \texttt{c}.(\texttt{a},\texttt{c}) \in \texttt{ocorre}^{\mathcal{I}} \wedge (\texttt{c},\texttt{b}) \in \texttt{pref}^{\mathcal{I}}\} \cap \\
&\quad \{(\texttt{a},\texttt{b}) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} | (\texttt{b},\texttt{a}) \in \texttt{lecciona}^{\mathcal{I}}\} = \\
&= \{(\texttt{a},\texttt{b}) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} | (\texttt{b},\texttt{a}) \in \texttt{lecciona}^{\mathcal{I}} \wedge \exists \texttt{c}.(\texttt{a},\texttt{c}) \in \texttt{ocorre}^{\mathcal{I}} \\
&\quad \wedge (\texttt{c},\texttt{b}) \in \texttt{pref}^{\mathcal{I}}\}
\end{aligned}$$

resulta que a semântica da expressão (2.1) é

$$\begin{aligned}
\{\texttt{a} \in \texttt{Disciplina}^{\mathcal{I}} | \forall \texttt{b}.(\texttt{b},\texttt{a}) \in \texttt{lecciona}^{\mathcal{I}} \wedge \exists \texttt{c}.(\texttt{a},\texttt{c}) \in \texttt{ocorre}^{\mathcal{I}} \\
\wedge (\texttt{c},\texttt{b}) \in \texttt{pref}^{\mathcal{I}} \Rightarrow \texttt{b} \in \texttt{Professor}^{\mathcal{I}}\}
\end{aligned}$$

▲

Conceitos	**Interpretação**
$\top$	$\Delta^{\mathcal{I}}$
$\bot$	$\emptyset$
$\mathtt{C} \sqcap \mathtt{D}$	$\mathtt{C}^{\mathcal{I}} \cap \mathtt{D}^{\mathcal{I}}$
$\mathtt{C} \sqcup \mathtt{D}$	$\mathtt{C}^{\mathcal{I}} \cup \mathtt{D}^{\mathcal{I}}$
$\neg\, \mathtt{C}$	$\Delta^{\mathcal{I}} \setminus \mathtt{C}^{\mathcal{I}}$
$\forall\, \mathtt{p}.\mathtt{C}$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \mathtt{p}^{\mathcal{I}}(\mathtt{d}) \subseteq \mathtt{C}^{\mathcal{I}}\}$
$\exists\, \mathtt{p}.\mathtt{C}$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \mathtt{p}^{\mathcal{I}}(\mathtt{d}) \cap \mathtt{C}^{\mathcal{I}} \neq \emptyset\}$
atmost$(\mathtt{n}, \mathtt{p})$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \#(\mathtt{p}^{\mathcal{I}}(\mathtt{d})) \leq \mathtt{n}\}$
atleast$(\mathtt{n}, \mathtt{p})$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \#(\mathtt{p}^{\mathcal{I}}(\mathtt{d})) \geq \mathtt{n}\}$
atmost$(\mathtt{n}, \mathtt{p}$ **is** $\mathtt{C})$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \#(\mathtt{p}^{\mathcal{I}}(\mathtt{d}) \cap \mathtt{C}^{\mathcal{I}}) \leq \mathtt{n}\}$
atleast$(\mathtt{n}, \mathtt{p}$ **is** $\mathtt{C})$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \#(\mathtt{p}^{\mathcal{I}}(\mathtt{d}) \cap \mathtt{C}^{\mathcal{I}}) \geq \mathtt{n}\}$
$\mathtt{p}$ **is** $\mathtt{b}$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \mathtt{b}^{\mathcal{I}} \in \mathtt{p}^{\mathcal{I}}(\mathtt{d})\}$
one_of$(\mathtt{b}_1, ..., \mathtt{b}_n)$	$\{{\mathtt{b}_1}^{\mathcal{I}}, ..., {\mathtt{b}_n}^{\mathcal{I}}\}$
sameas$(\mathtt{p}, \mathtt{q})$	$\{\mathtt{d} \in \Delta^{\mathcal{I}} \mid \mathtt{p}^{\mathcal{I}}(\mathtt{d}) = \mathtt{q}^{\mathcal{I}}(\mathtt{d})\}$
product$(\mathtt{C}, \mathtt{D})$	$\mathtt{C}^{\mathcal{I}} \times \mathtt{D}^{\mathcal{I}}$

Tabela 2.1: DL: Compêndio de construtores de conceitos.

A introdução de novos construtores altera, no entanto, as propriedades computacionais dos problemas de satisfazibilidade e subsunção associados às linguagens de descrição [32]. Nas Tabelas 2.1 e 2.2 são apresentados em resumo os construtores de conceitos e qualidades e suas interpretações.

2.3.3 Domínios Concretos

Nesta subsecção, estendemos a linguagem de descrição anterior a domínios concretos. Os domínios concretos permitem a integração de conceitos concretos, tais como números, intervalos de tempo e *strings* na linguagem de descrição. Para tal, começamos por definir formalmente o conceito de domínios concretos.

Definição 2.3.6 *Um domínio concreto $\mathcal{D}$ é um par ($\Delta^{\mathcal{D}}$,pred($\mathcal{D}$)), onde $\Delta^{\mathcal{D}}$ é um conjunto e pred($\mathcal{D}$) é um conjunto de nomes de predicados. A cada nome de predicado $\mathtt{P} \in pred(\mathcal{D})$ é associado uma aridade n de forma que $\mathtt{P}^{\mathcal{D}} \subseteq (\Delta^{\mathcal{D}})^n$. Sejam $\mathtt{u}_1, ..., \mathtt{u}_k$ sequências não vazias de atributos abstratos, um conceito concreto é identificado por $\mathtt{P}(\mathtt{u}_1, ..., \mathtt{u}_n)$.*

Os domínios concretos consistem num domínio, tal como o conjunto dos números inteiros, e num conjunto de predicados, tais como a qualidade "<" e a relação ternária "+". Enriquecendo a linguagem de descrição $\mathcal{ALC}$ com um domínio concreto $\mathcal{D}$, como o referido, obtemos a lógica de descrição básica com domínios concretos $\mathcal{ALC}(\mathcal{D})$ [76, 10, 8].

Mais precisamente, uma linguagem de descrição com adição dos domínios concretos $\mathcal{ALC}(\mathcal{D})$ é uma linguagem de descrição que se obtém de $\mathcal{ALC}$ por acréscimo de:

Qualidades	**Interpretação**
$\top$	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
Id	$\{(\mathtt{a},\mathtt{a})\|\mathtt{a} \in \Delta^{\mathcal{I}}\}$
$\mathtt{p} \sqcap \mathtt{q}$	$\mathtt{p}^{\mathcal{I}} \cap \mathtt{q}^{\mathcal{I}}$
$\mathtt{p} \sqcup \mathtt{q}$	$\mathtt{p}^{\mathcal{I}} \cup \mathtt{q}^{\mathcal{I}}$
$\neg\, \mathtt{p}$	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \setminus \mathtt{p}^{\mathcal{I}}$
inverse p	$\{(\mathtt{a},\mathtt{b})\|(\mathtt{b},\mathtt{a}) \in \mathtt{p}^{\mathcal{I}}\}$
sub(p, q)	$\{(\mathtt{a},\mathtt{b}) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}\|(\mathtt{a},\mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \Rightarrow (\mathtt{a},\mathtt{b}) \in \mathtt{q}^{\mathcal{I}}\}$
restrict(p, C)	$\{(\mathtt{a},\mathtt{b}) \in \mathtt{p}^{\mathcal{I}}\|\mathtt{b} \in \mathtt{C}^{\mathcal{I}}\}$
$\mathtt{p} \cdot \mathtt{q}$	$\mathtt{p}^{\mathcal{I}} \circ \mathtt{q}^{\mathcal{I}}$

Tabela 2.2: DL: Compêndio de construtores de qualidades.

- *atributos abstratos* - qualidades interpretadas como relações funcionais;
- *atributos concretos* - um novo tipo de nomes interpretados como uma função parcial do domínio lógico no domínio concreto;
- um novo construtor de conceitos que permite descrever restrições nos valores concretos usando predicados do domínio concreto.

Sejam $\mathtt{u} = \mathtt{f_1}...\mathtt{f_m}\ \mathtt{g}$ uma sequência não vazia de atributos abstratos $\mathtt{f_1}, ..., \mathtt{f_m}$ e g um atributo concreto, d um nome de indivíduo, tem-se que a semântica da sequência u é dada pela composição dos atributos abstratos e do atributo concreto, respeitando a ordem dos atributos na sequência. Ou seja,

$$\mathtt{u}^{\mathcal{I}}(\mathtt{d}) = \mathtt{g}^{\mathcal{I}}(\mathtt{f}^{\mathcal{I}}{}_{\mathtt{m}}(\mathtt{f}^{\mathcal{I}}{}_{\mathtt{m}-1}(...(\mathtt{f}^{\mathcal{I}}{}_{1}(\mathtt{d}))...))).$$

Uma qualidade num domínio concreto $\mathcal{D}$ é identificada pelo conceito $\mathtt{P}(\mathtt{u_1}, ..., \mathtt{u_n})$, cuja semântica é

$$\begin{aligned} \mathtt{P}(\mathtt{u_1}, ..., \mathtt{u_n})^{\mathcal{I}} &= \{\mathtt{d} \in \Delta^{\mathcal{I}}|(\mathtt{u_1}^{\mathcal{I}}(\mathtt{d}), ..., \mathtt{u_n}^{\mathcal{I}}(\mathtt{d})) \in \mathtt{P}^{\mathcal{D}}\} = \\ &= \{\mathtt{d} \in \Delta^{\mathcal{I}}|\exists \mathtt{x_1}, ..., \mathtt{x_n} \in \Delta^{\mathcal{D}} :\ \mathtt{u}^{\mathcal{I}}{}_{1}(\mathtt{d}) = \mathtt{x_1} \wedge ... \wedge \mathtt{u}^{\mathcal{I}}{}_{\mathtt{n}}(\mathtt{d}) = \mathtt{x_n} \\ &\quad \wedge\ (\mathtt{x_1}, ..., \mathtt{x_n}) \in \mathtt{P}^{\mathcal{D}}\}, \end{aligned}$$

e o novo construtor é identificado por g↑, cuja semântica é

$$(\mathtt{g}\uparrow)^{\mathcal{I}} = \{\mathtt{d} \in \Delta^{\mathcal{I}} : \mathtt{g}^{\mathcal{I}}(\mathtt{d})\ \mathtt{indefinido}\}$$

onde P é um predicado do domínio concreto, g um atributo concreto e $\mathtt{u_1}, ..., \mathtt{u_n}$ são sequências de atributos.

Normalmente, utiliza-se u↑ para abreviar $\forall \mathtt{f_1}...\forall \mathtt{f_k}\ \mathtt{g}\uparrow$ se $\mathtt{u} = \mathtt{f_1}...\mathtt{f_k}\ \mathtt{g}$.

Exemplo 2.3.7 Suponhamos que pretendemos afectar salas a determinadas disciplinas de acordo com a inscrição prévia dos alunos. Consideremos Sala um conceito primitivo,

`afecta` uma qualidade primitiva, `nalunos` um atributo abstrato que representa o número de alunos afetos a uma disciplina e `capacidade` um atributo abstrato que representa a capacidade de uma sala. O conceito

$$\texttt{Sala} \sqcap [\geq (\texttt{capacidade}, \texttt{afecta} \cdot \texttt{nalunos})]$$

exprime o conjunto de todas as salas onde é respeitada a capacidade. Ou seja,

$$\begin{aligned} & (\texttt{Sala} \sqcap [\geq (\texttt{capacidade}, \texttt{afecta} \cdot \texttt{nalunos})])^{\mathcal{I}} = \\ = \; & \{\texttt{a} \in \Delta^{\mathcal{I}} | \texttt{a} \in \texttt{Sala}^{\mathcal{I}}\} \cap \{\texttt{a} \in \Delta^{\mathcal{I}} | \exists \texttt{x}_1, \texttt{x}_2 \in \Delta^{\mathcal{D}} : \; \texttt{capacidade}^{\mathcal{I}}(\texttt{a}) = \texttt{x}_1 \\ & \wedge \, (\texttt{afecta} \cdot \texttt{nalunos})^{\mathcal{I}}(\texttt{a}) = \texttt{x}_2 \wedge \; (\texttt{x}_1, \texttt{x}_2) \in \geq^{\mathcal{D}}\} = \\ = \; & \{\texttt{a} \in \texttt{Sala}^{\mathcal{I}} | \exists \texttt{x}_1, \texttt{x}_2 \in \mathbb{Q}, \texttt{c} \in \Delta^{\mathcal{I}} : (\texttt{a}, \texttt{c}) \in \texttt{afecta}^{\mathcal{I}} \wedge \\ & \texttt{capacidade}^{\mathcal{I}}(\texttt{a}) = \texttt{x}_1 \geq \texttt{x}_2 = \texttt{nalunos}^{\mathcal{I}}(\texttt{c})\}. \end{aligned}$$

▲

A função de interpretação faz a correspondência entre `f`, um atributo abstrato, e uma função parcial $\texttt{f}^{\mathcal{I}}$ de $\Delta^{\mathcal{I}}$ em $\Delta^{\mathcal{I}}$. E a cada atributo concreto `g` corresponde uma função parcial $\texttt{g}^{\mathcal{I}}$ do domínio $\Delta^{\mathcal{I}}$ no domínio concreto $\Delta^{\mathcal{D}}$.

Definição 2.3.7 *Seja $\mathcal{D}$ um domínio concreto e $\mathcal{V}$ um conjunto de variáveis. Uma conjunção-$\mathcal{D}$ é um predicado da forma*

$$\mathbf{c} = \bigwedge_{\texttt{i}=1}^{k} \texttt{P}_\texttt{i}(\texttt{x}_0^{(\texttt{i})}, ..., \texttt{x}_{\texttt{n}_\texttt{i}}^{(\texttt{i})}),$$

onde $\texttt{P}_\texttt{i}$ é um predicado de aridade $\texttt{n}_\texttt{i}$ para $\texttt{i} = 1, ..., \texttt{k}$ e $\texttt{x}_\texttt{j}^{(\texttt{i})}$ são elementos de $\mathcal{V}$. Uma conjunção-$\mathcal{D}$ $\mathbf{c}$ é satisfazível se e só se existe uma função δ que às variáveis de $\mathbf{c}$ faz corresponder elementos de $\Delta^{\mathcal{D}}$ de forma que $(\delta(\texttt{x}_0^{(\texttt{i})}), ..., \delta(\texttt{x}_{\texttt{n}_\texttt{i}}^{(\texttt{i})})) \in \texttt{P}_\texttt{i}{}^{\mathcal{D}}$ para cada $\texttt{i} = 1, ..., \texttt{k}$. Tal função é denotada de solução de $\mathbf{c}$. Diz-se que o domínio concreto $\mathcal{D}$ é admissível se e só se:

(i) o seu conjunto de nomes de predicados é fechado sob a negação, ou seja, $pred(\mathcal{D})$ contém, para cada predicado $\texttt{P} \in pred(\mathcal{D})$ de aridade $\texttt{n}$, um predicado $\bar{\texttt{P}} \in pred(\mathcal{D})$ de aridade $\texttt{n}$ com $\bar{\texttt{P}}^{\mathcal{D}} = \Delta^{\mathcal{D}} \setminus \texttt{P}^{\mathcal{D}}$,

(ii) $pred(\mathcal{D})$ contém o predicado $\top^{\mathcal{D}}$ para $\Delta^{\mathcal{D}}$ e

(iii) a satisfazibilidade da conjunção-$\mathcal{D}$ finita sobre $pred(\mathcal{D})$ é decidível, ou seja, a satisfazibilidade da fórmula $\texttt{P}_1(\texttt{x}_0^{(1)}, ..., \texttt{x}_{\texttt{n}_1}^{(1)}) \wedge ... \wedge \texttt{P}_\texttt{k}(\texttt{x}_0^{(\texttt{k})}, ..., \texttt{x}_{\texttt{n}_\texttt{k}}^{(\texttt{k})})$ é decidível, onde $\texttt{P}_\texttt{i}$ são predicados de aridade $\texttt{n}_\texttt{i}$.

	Complexidade da Satisfazibilidade de $\mathcal{D}$	Complexidade da Satisfazibilidade dos Conceitos $\mathcal{ALC}(\mathcal{D})$
Q + " $*$ " + "`int`"	indecidível	indecidível
Q + " $*$ "	em EXPTIME	em NEXPTIME
Q + "`int`"	NP-completo	PSPACE-completo
Q	em PTIME	PSPACE-completo

Tabela 2.3: Domínios concretos numéricos e respetivas complexidades.

2.3.3.1 Domínio Concreto Numérico $\mathbb{Q}$

Um domínio concreto numérico muito utilizado é o conjunto dos números racionais $\mathbb{Q}$. De acordo com a definição de domínio concreto temos como domínio $\Delta^{\mathbb{Q}}$ o conjunto dos racionais $\mathbb{Q}$. Os predicados associados a este conjunto são os seguintes:

- predicados unários $\mathtt{P}_{\mathtt{q}}$, para cada $\mathtt{P} \in \{<, \leq, =, \neq, \geq, >\}$ e cada $\mathtt{q} \in \mathbb{Q}$ com $(\mathtt{P}_{\mathtt{q}})^{\mathrm{Q}} = \{\mathtt{q}' \in \mathbb{Q} | \mathtt{q}' \ \mathtt{P} \ \mathtt{q}\}$;
- predicados binários $<, \leq, =, \neq, \geq, >$, com a extensão óbvia;
- predicados ternários $+$ e $\mp$ com $(+)^{\mathrm{Q}} = \{(\mathtt{q}, \mathtt{q}', \mathtt{q}'') \in \mathbb{Q}^3 | \mathtt{q} + \mathtt{q}' = \mathtt{q}''\}$ e $(\mp)^{\mathrm{Q}} = \mathbb{Q}^3 \setminus (+)^{\mathrm{Q}}$;
- o predicado unário $\top^{\mathrm{Q}}$ com $(\top^{\mathrm{Q}})^{\mathrm{Q}} = \mathbb{Q}$ e o predicado $\bot^{\mathrm{Q}}$ com $(\bot^{\mathrm{Q}})^{\mathrm{Q}} = \emptyset$.

A presença dos predicados $\top^{\mathrm{Q}}$ com $\bot^{\mathrm{Q}}$ e a negação do predicado "+", "-", estão relacionados com a admissibilidade dos domínios concretos.

Exemplo 2.3.8 Consideremos `Aula` um conceito primitivo e `duracao` um atributo abstrato (relação funcional), então a inclusão do domínio concreto Q permite-nos construir conceitos tais como

$$\mathtt{Aula} \sqcap \leq (\mathtt{duracao}, 2)$$

que exprime todos as aulas que têm duração não superior a 2 horas. ▲

Existe um vasto conjunto de predicados que podem ser acrescentados ao domínio concreto Q de forma a estender a sua expressividade. Os mais usuais são:

- os predicados ternários $*$ e $\overline{*}$ com $(*)^{\mathrm{Q}} = \{(\mathtt{q}, \mathtt{q}', \mathtt{q}'') \in \mathbb{Q}^3 | \mathtt{q}.\mathtt{q}' = \mathtt{q}''\}$ e $(\overline{*})^{\mathbb{Q}} = \mathbb{Q}^3 \setminus (*)^{\mathrm{Q}}$;
- os predicados unários `int` e $\overline{\mathtt{int}}$ com $(\mathtt{int})^{\mathrm{Q}} = \mathbb{Z}$ (onde $\mathbb{Z}$ denota o conjunto dos números inteiros) e $(\overline{\mathtt{int}})^{\mathrm{Q}} = \mathbb{Q} \setminus \mathbb{Z}$.

Adicionando diferentes combinações destes predicados, obtemos três extensões do domínio concreto Q, apresentadas na Tabela 2.3, com os conhecidos valores de complexidade da satisfazibilidade-$\mathcal{D}$ e da satisfazibilidade dos conceitos de $\mathcal{ALC}(\mathcal{D})$ [76].

2.3.3.2 Intervalos de Tempo

As linguagens de descrição básicas estão concentradas na representação de conceitos estáticos, mas recentemente tem vindo a aumentar o interesse pelas linguagens de descrição mais dinâmicas que permitam incorporar aspetos temporais do domínio de aplicação. Uma das possibilidades de incorporar os intervalos de tempo como primitivas temporais é usar as relações temporais de Allen [3]. O poder expressivo das linguagens de descrição com intervalos de tempo é usualmente baseado em construtores de conceitos que se referem às 13 relações de Allen, ver Figura 2.1, que descrevem todas as possibilidades de como dois intervalos de tempo podem ser relacionados entre si. Embora estas linguagens tenham uma forte expressividade, é bastante difícil evitar que se tornem indecidíveis.

Vamos, de seguida, expor a lógica de descrição de decisão temporal $\mathcal{TDL}$ [8, 77] que admite TBox gerais e permite representações e raciocínios sobre os intervalos de tempo. De facto, $\mathcal{TDL}$ admite a representação das relações entre intervalos de tempo de Allen. Mais precisamente, $\mathcal{TDL}$ é uma extensão da lógica de descrição básica $\mathcal{ALC}$, incluindo:

- TBox;
- *atributos abstratos* - qualidades interpretadas como relações funcionais;
- *atributos temporais* - nomes de qualidades que permitem associar instantes temporais (números racionais) com elementos do domínio;
- *um construtor de conceitos temporal* - permite estabelecer se dois instantes de tempo descritos através de atributos temporais estão numa das relações $<, \leq, =, \neq, \geq, >$.

Desta forma, $\mathcal{TDL}$ pode ser vista como a extensão de $\mathcal{ALC}$ com TBox e com um domínio concreto particular, $\mathcal{TDL}(\mathcal{D}) = (\mathbb{Q}, \{<, \leq, =, \neq, \geq, >\})$, onde todos os predicados são qualidades com a semântica óbvia.

Exemplo 2.3.9 O seguinte conceito $\mathcal{TDL}$ define as aulas que ocorrem antes do final do bloco da manhã

$$\texttt{Aula} \sqcap \leq (\texttt{fim}, \texttt{manha} \cdot \texttt{fim}).$$

Nesta condição, `Aula` é um conceito primitivo, `manha` é um atributo abstrato, e `fim` é um atributo temporal cujo valor é o racional que corresponde ao término da aula - um instante no espaço de tempo. O conceito $\texttt{manha} \cdot \texttt{fim}$ é a composição do atributo abstrato `manha` com o atributo temporal `fim` e que corresponde ao instante de tempo onde termina o atributo `manha`. O conceito apresentado devolve as aulas cujo fim é menor ou igual ao fim do atributo `manha`. ▲

A melhor forma de estabelecer as relações temporais de Allen é representar os intervalos de tempo em termos dos seus instantes de início e fim e depois usar as relações sobre os instantes de tempo $<, \leq, =, \neq, \geq, >$.

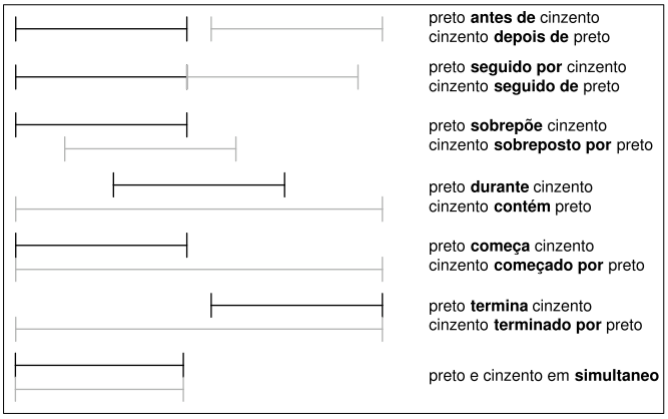


Figura 2.1: As relações temporais de Allen.

Exemplo 2.3.10 Se pretendermos saber quais as aulas que estão contidas num intervalo de tempo correspondente ao bloco `manha`, podemos escrever

$$\texttt{Aula} \sqcap [\geq (\ell, \texttt{manha} \cdot \ell)] \sqcap [\leq (\texttt{r}, \texttt{manha} \cdot \texttt{r})]$$

$$\sqcap [< (\ell, \texttt{r})] \sqcap [< (\texttt{manha} \cdot \ell, \texttt{manha} \cdot \texttt{r})]$$

onde o atributo temporal ℓ representa o extremo esquerdo (instante inicial) do intervalo de tempo e o atributo temporal `r` o extremo direito (instante final) do intervalo de tempo.

▲

A estrutura da representação consiste num conjunto de convenções e abreviaturas. Assumindo que todas as entidades do domínio de aplicação ou são entidades temporais (`Temporal`) ou atemporais (`ATemporal`). Consideremos `t` um atributo temporal que representa um instante no tempo, ℓ o atributo temporal que representa o extremo esquerdo (instante início) do intervalo de tempo e o atributo temporal `r` o extremo direito (instante final) do intervalo de tempo. A respetiva TBox $\mathcal{T}$ está expressa na Figura 2.2. Os primeiros quatro conceitos definem as noções relevantes enquanto que o quinto conceito traduz os casos problemáticos onde os objetos podem ser instantes de tempo ou intervalos de tempo. De notar que a expressão $= (\texttt{t}, \texttt{t})$ apenas é usada para garantir que existe um valor associado para o atributo temporal `t`. Os conceitos `ATemporal`, `Instante` e `Intervalo` são mutuamente disjuntos, e a sua união é equivalente a $\top$, ou seja, ao domínio $\Delta^{\mathcal{I}}$.

O raciocínio das linguagens de descrição com adição dos intervalos de tempo com $\mathcal{TDL}$ é baseado nas relações temporais de Allen, indicadas na Figura 2.1. Para tornar os conceitos legíveis, define-se uma agradável abreviatura para cada uma das relações.

Exemplo 2.3.11 Em particular, o conceito

$$\textbf{contem}(\texttt{u}, \texttt{u}') \text{ abrevia a relação } [< (\texttt{u} \cdot \ell, \texttt{u}' \cdot \ell) \sqcap > (\texttt{u} \cdot \texttt{r}, \texttt{u}' \cdot \texttt{r})].$$

Bem como o conceito

$$\textbf{seguido}(\mathtt{u}, \mathtt{u}') \text{ abrevia a relação } [= (\mathtt{u} \cdot \mathtt{r}, \mathtt{u}' \cdot \ell)].$$

Onde $\mathtt{u}$ e $\mathtt{u}'$ são sequências de atributos abstratos.

Consideremos o atributo abstrato `self` para denotar uma sequência vazia de atributos abstratos. Assim,

$$\textbf{comeca}(\mathtt{u}, \mathtt{self}) \text{ abrevia a relação } [= (\mathtt{u} \cdot \ell, \ell) \sqcap < (\mathtt{u} \cdot \mathtt{r}, \mathtt{r})].$$

Intuitivamente, `self` refere-se ao intervalo de tempo associado ao elemento do domínio no qual o conceito **comeca**$(\mathtt{u}, \mathtt{self})$ é atribuído. Da mesma forma

$$\textbf{termina}(\mathtt{u}, \mathtt{self}) \text{ abrevia a relação } [> (\mathtt{u} \cdot \ell, \ell) \sqcap = (\mathtt{u} \cdot \mathtt{r}, \mathtt{r})].$$

▲

O domínio concreto constituído pelas relações temporais é traduzido por instantes de tempo ou intervalos de tempo. Desta forma, é possível estabelecer cinco relações temporais entre instantes de tempo e intervalos de tempo, cujas abreviaturas são:

$$\begin{array}{rll}
\textbf{antesp}(\mathtt{u}, \mathtt{u}') & \text{para} & < (\mathtt{u} \cdot \mathtt{t}, \mathtt{u}' \cdot \ell) \\
\textbf{comecap}(\mathtt{u}, \mathtt{u}') & \text{para} & = (\mathtt{u} \cdot \mathtt{t}, \mathtt{u}' \cdot \ell) \\
\textbf{durantep}(\mathtt{u}, \mathtt{u}') & \text{para} & < (\mathtt{u}' \cdot \ell, \mathtt{u} \cdot \mathtt{t}) \sqcap < (\mathtt{u} \cdot \mathtt{t}, \mathtt{u}' \cdot \mathtt{r}) \\
\textbf{terminap}(\mathtt{u}, \mathtt{u}') & \text{para} & = (\mathtt{u} \cdot \mathtt{t}, \mathtt{u}' \cdot \mathtt{r}) \\
\textbf{depoisp}(\mathtt{u}, \mathtt{u}') & \text{para} & < (\mathtt{u}' \cdot \mathtt{r}, \mathtt{u} \cdot \mathtt{t})
\end{array}$$

onde $\mathtt{u}$ e $\mathtt{u}'$ são sequências de atributos abstratos, $\ell, \mathtt{r}$ e $\mathtt{t}$ atributos temporais.

2.3.4 Funções de Agregação

Nesta subsecção vamos introduzir uma extensão de $\mathcal{ALC}(\mathcal{D})$ com agregação [10, 76]. As funções de agregação são um mecanismo disponível em muitos formalismos de representação, tais como as base de dados e as linguagens de interrogação sobre as bases de dados, e servem para expandir as linguagens de descrição aos domínios concretos.

Antes de definirmos as funções de agregação, vamos definir o conceito de multi-conjunto: ao contrário dos conjuntos simples, cada elemento pode ocorrer várias vezes num multi-conjunto, mas num número finito de vezes. Por exemplo, no multi-conjunto $\{a, a, b, c, c, c\}$ os elementos a, b e c ocorrem respetivamente 2, 1 e 3 vezes. Os multi-conjuntos são necessários para garantir a distinção dos dados quando ocorrem diversas vezes.

Definição 2.3.8 *Um multi-conjunto é um par (S, M), onde S é um conjunto finito e $M : S \to \mathbb{N}$ é a aplicação dos elementos de S no conjunto dos números naturais. Para cada $s \in S$, $M(s)$ é o número de vezes que s ocorre em M. Um multi-conjunto (S, M) é dito finito se $\{s|M(s) \neq 0\}$ é um conjunto finito.*

Como as funções de agregação dependem de um domínio concreto específico, a noção de domínio concreto é estendida de acordo com esse facto. Informalmente, um atributo concreto ou é um atributo, ou é uma sequência de atributos, ou é obtido por aplicação de uma função de agregação à composição de uma qualidade com um atributo.

Definição 2.3.9 *Um domínio concreto com agregação consiste num domínio concreto, definido previamente na Definição 2.3.6, enriquecido de um conjunto de funções de agregação $agreg(\mathcal{D})$, onde a cada $\Gamma \in agreg(\mathcal{D})$ é associado uma função parcial $\Gamma^{\mathcal{D}}$ do conjunto finito de multi-conjuntos de $\Delta^{\mathcal{D}}$ em $\Delta^{\mathcal{D}}$. Para indicar que um domínio concreto dispõe de funções de agregação passamos a denominá-lo por Σ.*

Finalmente, os conceitos de $\mathcal{ALC}(\Sigma)$ são obtidos dos conceitos de $\mathcal{ALC}(\mathcal{D})$ e em adição, permitem a utilização de atributos concretos $\mathtt{f_i}$ nos predicados $\mathtt{P}(\mathtt{f_1}, ..., \mathtt{f_n})$ (em $\mathcal{ALC}(\mathcal{D})$ apenas é permitido cadeias de atributos abstratos).

Para definir a semântica de atributos de agregação, utilizamos o multi-conjunto $(\Delta^{\Sigma}, \mathtt{M}_{\mathtt{a}}^{\mathtt{p}\cdot\mathtt{f}})$, onde a aplicação $\mathtt{M}_{\mathtt{a}}^{\mathtt{p}\cdot\mathtt{f}}$ faz a correspondência entre cada elemento $\mathtt{z} \in \Delta^{\Sigma}$ e o número de ocorrências de $\mathtt{p}$-sucessores do indivíduo $\mathtt{a}$ que têm $\mathtt{z}$ como $\mathtt{f}$-sucessor (onde $\mathtt{p}$ é uma qualidade e $\mathtt{f}$ um atributo):

$$\mathtt{M}_{\mathtt{a}}^{\mathtt{p}\cdot\mathtt{f}}(\mathtt{z}) = \#\{\mathtt{b} \in \Delta^{\mathcal{I}} | (\mathtt{a}, \mathtt{b}) \in \mathtt{p}^{\mathcal{I}} \wedge \mathtt{f}^{\mathcal{I}}(\mathtt{b}) = \mathtt{z}\}.$$

Ou seja, $\mathtt{M}_{\mathtt{a}}^{\mathtt{p}\cdot\mathtt{f}}(\mathtt{z})$ é o número de vezes que ocorre a asserção $\mathtt{p}(\mathtt{a}, \mathtt{b})$ quando se tem o atributo $\mathtt{f}(\mathtt{b}, \mathtt{z})$ na interpretação $\mathcal{I}$.

E finalmente, a semântica dos atributos de agregação é definida como:

$$(\mathtt{f_1}...\mathtt{f_m}\ \Gamma(\mathtt{p}\cdot\mathtt{f}))^{\mathcal{I}}(\mathtt{a}) = \begin{cases} \Gamma^{\Sigma}(\mathtt{M}_{\mathtt{a}'}^{\mathtt{p}\cdot\mathtt{f}}) & \mathtt{se}\ (\mathtt{f_1}...\mathtt{f_m})^{\mathcal{I}}(\mathtt{a}) = \mathtt{a}' \in \Delta^{\mathcal{I}} \\ \mathtt{indefinido} & \mathtt{se}\ (\mathtt{f_1}...\mathtt{f_m})^{\mathcal{I}}(\mathtt{a}) \notin \Delta^{\mathcal{I}} \end{cases}$$

e $\Gamma^{\Sigma}(\mathtt{M}_{\mathtt{a}'}^{\mathtt{p}\cdot\mathtt{f}})$ é chamado de $(\mathtt{f_1}...\mathtt{f_m}\ \Gamma(\mathtt{p}\cdot\mathtt{f}))$-sucessor de $\mathtt{a}$.

Como consequências desta definição temos:

- Se o indivíduo $\mathtt{a}$ tiver um $\mathtt{p}$-sucessor $\mathtt{b}$ com um atributo funcional $\mathtt{f}$-sucessor, então o indivíduo $\mathtt{b}$ não tem influência em $\mathtt{M}_{\mathtt{a}}^{\mathtt{p}\cdot\mathtt{f}}$: é definido tendo apenas em conta $(\mathtt{p}\cdot\mathtt{f})$-sucessores de $\mathtt{a}$ no domínio concreto Δ^{Σ}.

$$\begin{aligned}
\texttt{ATemporal} &\equiv \texttt{t}\uparrow \sqcap \ell\uparrow \sqcap \texttt{r}\uparrow \\
\texttt{Temporal} &\equiv \texttt{Instante} \sqcup \texttt{Intervalo} \\
\texttt{Instante} &\equiv =(\texttt{t},\texttt{t}) \\
\texttt{Intervalo} &\equiv <(\ell,\texttt{r}) \\
\top &\equiv (\neg =(\texttt{t},\texttt{t}) \sqcup (\ell\uparrow \sqcap \texttt{r}\uparrow)) \sqcap (\neg (=(\ell,\ell) \sqcup =(\texttt{r},\texttt{r}))) \\
&\quad \sqcup (<(\ell,\texttt{r}) \sqcap \texttt{t}\uparrow)
\end{aligned}$$

Figura 2.2: TBox $\mathcal{T}$ com definições básicas de entidades.

- Como $\Delta^{\mathcal{I}}$ é finito, cada $(\Delta^{\mathcal{I}}, \Gamma^{\Sigma}(\mathtt{M}^{\mathtt{p}\,\cdot\,\mathtt{f}}_{\mathtt{a}'}))$ é necessariamente um multi-conjunto finito. No entanto, existem duas razões para os quais $(\mathtt{f}_1 ... \mathtt{f}_\mathtt{m}\ \Gamma(\mathtt{p}\cdot\mathtt{f}))^{\mathcal{I}}(\mathtt{a})$ possa não estar definido: primeiro, $\mathtt{a}$ pode não ter $(\mathtt{f}_1 ... \mathtt{f}_\mathtt{m})$-sucessor $\mathtt{a}' \in \Delta^{\mathcal{I}}$; segundo, as funções de agregação podem ser parciais.

De seguida, apresentamos as funções de agregação `sum`, `count`, `min` e `max` que são definidas para os multi-conjuntos $(\mathbb{Q}, \mathtt{M})$, onde $\mathbb{Q}$ é o conjunto dos números racionais:

$$\begin{aligned}
\mathtt{sum}(\mathtt{M}) &= \sum_{\mathtt{y}\in \mathtt{M}} \mathtt{M}(\mathtt{y})\mathtt{y} \\
\mathtt{count}(\mathtt{M}) &= \sum_{\mathtt{y}\in \mathtt{M}} \mathtt{M}(\mathtt{y}) \\
\mathtt{min}(\mathtt{M}) &= \begin{cases} \mathtt{m} & \mathtt{se}\ \exists \mathtt{m}\in \mathtt{M} : \mathtt{n} \geq \mathtt{m}, \forall \mathtt{n} \in \mathtt{M} \\ \mathtt{indefinido} & \mathtt{c.c.} \end{cases} \\
\mathtt{max}(\mathtt{M}) &= \begin{cases} \mathtt{m} & \mathtt{se}\ \exists \mathtt{m}\in \mathtt{M} : \mathtt{n} \leq \mathtt{m}, \forall \mathtt{n} \in \mathtt{M} \\ \mathtt{indefinido} & \mathtt{c.c.} \end{cases}
\end{aligned}$$

Exemplo 2.3.12 O conjunto de funções de agregação introduzido permite-nos expressar conceitos tais como: "A soma das durações das aulas leccionados"

$$\mathbf{sum}(\mathtt{lecciona}\cdot\mathtt{duracao})$$

onde `lecciona` é uma qualidade e `duracao` é um atributo concreto. Ou expressar conceitos como "Os professores que têm uma carga horária total não superior ao pré-definido $MaxCH$"

$$\mathtt{Professor} \sqcap [\leq (\mathbf{sum}(\mathtt{lecciona}\cdot\mathtt{duracao}), \mathtt{MaxCH})]$$

onde `Professor` é um conceito primitivo e `MaxCH` é a constante que define a carga horária que pode ser atribuída aos professores. ▲

Na próxima secção iremos apresentar alguns dos sistemas computacionais para as linguagens de descrição.

2.3.5 Sistemas Computacionais para Lógicas Descritivas

Os sistemas computacionais Fact (*FAct Classification of Terminologies* [51, 96]), DLP (*Description Logic Prover* [96]) e Racer (*Renamed ABox and Concept Expression Reasoner* [43, 42]) fazem parte da primeira geração de sistemas otimizados de grande expressividade, que suportam subconjuntos de lógicas descritivas [92]. Inspirados em avanços teóricos - para assegurar restrições numéricas, conjunções de qualidades, generalização de inclusão de conceitos, axiomas cíclicos com semânticas descritivas, qualidades transitivas, hierarquias de relações e atributos, relações inversas, restrições numéricas quantificáveis e hierarquias de qualidades - esta nova geração de sistemas completos da lógica descritiva começa a desenvolver-se a partir do final da década de 1990. Atualmente, os projetos de suporte aos sistemas Fact e Racer já não estão ativos.

Inicialmente, a investigação em implementações práticas de sistemas computacionais para lógicas descritivas expressivas começou apenas com a análise dos conceitos e da terminologia, TBox. No entanto, mais do que implementar em cálculo de *tableaux* [94, 78] usado para demonstrar que as teorias são decidíveis e para a análise da complexidade, foi feita uma rigorosa investigação nos métodos de pesquisa de forma a tornarem os sistemas mais eficientes. Em particular, técnicas de otimização de casos médios foram desenvolvidos com o sistema Fact.

A primeira versão do sistema Fact suportava o raciocínio TBox para a lógica descritiva $\mathcal{ALCHF}(\mathbb{R}^+)$ e uma versão mais recente suportava o raciocínio TBox estendido às qualidades inversas e às restrições numéricas quantificáveis. O sistema Fact não suporta ABoxes. Em 2005, Ian Horrocks[7] (autor do sistema Fact) inserido no projeto *WonderWeb*[8] desenvolveu o sistema FaCT++. Este sistema é uma evolução do inicial Fact e suporta linguagens de ontologias baseadas em linguagens descritivas - OWL[9] (*Ontology Web Language)* e OWL2[10] (exceto *key constraint* e alguns tipos de dados).

O sistema DLP[11] é um sistema de lógicas descritivas que tem por base técnicas semelhantes às usadas no sistema Fact, embora utilize técnicas de otimização mais abrangentes. O DLP suporta conceitos de raciocínio consistente para a lógica de descrição $\mathcal{ALCN}(reg)$. O sistema DLP também não suporta as ABoxes, apenas a TBox.

Para muitas aplicações, para além da verificação da consistência dos conceitos e do raciocínio da TBox, o raciocínio da ABox também é fundamental. Um cálculo de tableau para a verificação da consistência da ABox foi apresentada para as seguintes linguagens: $\mathcal{ALCNH}(\mathbb{R}^+)$ e $\mathcal{ALCQHI}(\mathbb{R}^+)$, mais conhecida por $\mathcal{SHIQ}$ [75]. Baseado em resultados teóricos, uma implementação prática do cálculo de tableaux da ABox foi desenvolvida no sistema Racer, sistema de lógica descritiva bem mais completo que os referidos anterior-

[7] `http://www.comlab.ox.ac.uk/people/ian.horrocks/`
[8] `http://wonderweb.semanticweb.org/`
[9] `http://www.w3.org/TR/owl-ref/`
[10] `http://www.w3.org/TR/owl2-overview/`
[11] `http://www.bell-labs.com/user/pfps/dlp/`

mente, DLP e Fact. Ou seja, reconhece tanto o raciocínio da TBox como o da ABox. O sistema Racer suporta todas as técnicas de otimização incluídas no Fact. Fazem ainda parte deste sistema novas técnicas de otimização para lidar com restrições numéricas e com a ABox. A única regra imposta para as asserções da ABox é a unicidade dos nomes individuais.

As primeiras versões do Racer suportavam as lógicas $\mathcal{ALCNRH}(\mathbb{R}^+)$. Em versões mais recentes, o raciocínio foi estendido ao raciocínio da ABox com a lógica $\mathcal{SHIQ}(\mathbb{R}^+)$. Esta representação não é mais que a lógica descritiva básica $\mathcal{ALC}$ estendida às restrições numéricas quantificáveis, hierarquias de qualidades, qualidade inversa e qualidades transitivas. Em adição a estes atributos básicos, o sistema Racer também suporta domínios concretos sem as cadeias de atributos, para lidar com: restrições de máximo e mínimo sobre os inteiros, equações (inequações) polinomiais lineares sobre os reais ou cardinais com relações de ordem, equações (inequações) polinomiais não lineares multi-variadas sobre os números complexos, igualdades e desigualdades de *strings*. O sistema Racer suporta a especificação geral dos axiomas da terminologia. A TBox pode conter conceitos gerais de inclusão, que traduzem relação de subsunção entre os conceitos de dois termos. Múltiplas definições ou até mesmo definições cíclicas de conceitos são suportados pelo sistema Racer, ou seja, o sistema Racer suporta múltiplas TBoxes e ABoxes. A evolução deste sistema deu origem ao atual RacerPro [12]. Este sistema é caracterizado pela sua arquitetura em camadas (tipo cebola). O kernel do raciocínio situado no meio é o núcleo do sistema e implementa um *Tableau Calculus*[13] altamente otimizada para a decisão da consistência da ABox do problema da descrição lógica $\mathcal{SHIQ}(\mathcal{D}-)$. Alguns procedimentos especiais de inferências altamente otimizados, para linguagens de $\mathcal{SHIQ}(\mathcal{D})$, são também integrados no kernel e aplicados automaticamente a quando da entrada do problema, de forma a maximizar o seu desempenho.

2.4 *Resource Description Framework* (RDF)

A *Resource Description Framework* (RDF) [91, 25] é uma família de especificações criadas pela W3C que fornece um modelo de dados para anotações na Web Semântica, permitindo a partilha de dados na web. Essencialmente, a especificação RDF é uma linguagem que tem vindo a ser utilizada como um método geral para a descrição conceptual ou modelação de informação que é implementado em recursos da web, utilizando uma variedade de notações de sintaxe e formatos de dados de serialização.

O desenvolvimento da linguagem RDF tem sido motivada por diversos razões de utilização da web de dados, entre eles:

- Metadados web - fornece informação sobre recursos web e os sistemas que os usam

[12] `http://www.racer-systems.com/index.phtml`

[13] Um *Tableau Calculus* é um procedimento de decisão (conjunto de regras) para lógicas sentenciais e afins, e ainda um procedimento de prova para as fórmulas da lógica de primeira ordem. O *Tableau Calculus* também pode determinar a satisfazibilidade de conjuntos finitos de fórmulas de várias lógicas.

(por exemplo, classificação de conteúdos, descrições de capacidades, preferências de privacidades, etc.);

- Aplicações que necessitam modelos de informação abertos (por exemplo, horários de atividades, descrição de processos organizacionais, anotações nos recursos web, etc.);
- Auxiliar as máquinas no processamento da informação - permitir que os dados sejam processados fora do ambiente particular em que foram criados e de uma forma que possam ser trabalhados à escala da web;
- Interoperabilidade entre as aplicações: combinação de dados de várias aplicações para chegar a novas informações;
- Tratamento automático da informação web por agentes de software - a web está a evoluir no sentido de se ter a informação não apenas legível pelas pessoas, mas uma rede mundial de processos cooperativos. A linguagem RDF fornece uma representação adequada para estes processos.

A linguagem RDF foi projetada para representar a informação numa forma minimamente restrita e flexível: pode ser usada em aplicações isoladas, onde os formatos personalizados individualmente podem ser mais diretos e mais facilmente compreendidos. No entanto, a generalidade da utilização da linguagem RDF consiste na partilha da informação. O valor da informação aumenta a partir do momento que se torna acessível a um maior número de aplicações na web.

A linguagem RDF codifica o significado num conjunto de declarações. Cada declaração estabelece um facto sobre um recurso. Uma declaração RDF (também conhecido por triplo RDF) contém três componentes:

- o sujeito - é um recurso sobre o qual pretendemos estabelecer um facto;
- o predicado - é a propriedade que pretendemos declarar sobre o sujeito;
- o objeto - é o valor estabelecido na declaração e que pode ser um recurso, um literal ou um nó em branco (valor vazio).

Assim, a linguagem RDF descreve relações entre recursos em termos de propriedades e valores.

Um triplo RDF é convencionalmente escrito na ordem sujeito, predicado, objeto e tem a seguinte representação:

sujeito predicado objeto

A linguagem RDF representa os recursos da web em termos de propriedades anotadas. Os valores de propriedades anotadas (objetos, por exemplo) podem ser referências *Uniform*

Resource Identifier (URI)[14] de recursos da web ou literais (representações de valores de dados, tais como inteiros e strings). Um modelo RDF é representado por um conjunto de declarações.

Exemplo 2.4.1 Suponhamos que pretendemos representar através de um triplo RDF a seguinte declaração "Portugal's capital is Lisbon". O primeiro recurso "Portugal" é identificado com a referência URI `http://dbpedia.org/resource/Portugal` (sujeito do triplo). O segundo recurso "capital" é identificado com a referência URI `http://dbpedia.org/ontology/capital` (predicado do triplo) e o último recurso (objeto do triplo) é a identificação URI `http://dbpedia.org/resource/Lisbon` para o termo "Lisbon". Assim o seguinte triplo representa a declaração referida

```
<http://dbpedia.org/resource/Portugal>
      <http://dbpedia.org/ontology/capital>
            <http://dbpedia.org/resource/Lisbon> .
```

▲

Para representar as declarações RDF de forma a serem processadas pelas máquinas, a linguagem RDF utiliza a sintaxe específica *eXtensible Markup Language* (XML)[15], referida como RDF/XML[16]. Os recursos anotados RDF são normalmente identificados utilizando identificações URI. Para além deste identificador, os recursos web estão marcados com uma referência de localização na rede *Uniform Resource Locators* (URLs)[17].

Exemplo 2.4.2 A representação RDF/XML do exemplo anterior é estabelecida por:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:ontology="http://dbpedia.org/ontology/">
  <rdf:Description rdf:about="http://dbpedia.org/resource/Portugal">
    <ontology:capital rdf:resource="http://dbpedia.org/resource/Lisbon" />
  </rdf:Description>
</rdf:RDF>
```

▲

[14] Uma referência URI é uma sequência de caracteres usada para identificar um nome ou um recurso web. O recurso web `http://www.w3.org/` é o exemplo de uma referência URI (`http://www.w3.org/Addressing/URL/URI_Overview.html`).

[15] XML é uma linguagem de marcação que define um conjunto de regras para a codificação de documentos na web num formato que é simultaneamente legível por máquinas e pessoas (`http://www.w3.org/XML/`).

[16] `http://www.w3.org/TR/rdf-syntax-grammar/`

[17] `http://www.w3.org/Addressing/URL/`

2.5 *Ontology Web Language* (OWL)

Na Web Semântica, as ontologias definem os conceitos e relações (também conhecidos por termos) usados para descrever e representar um domínio de interesse. As ontologias são utilizadas para classificar os termos que poderão vir a ser usados (como domínio da aplicação) numa aplicação específica, caracteriza possíveis relações entre eles, e define possíveis restrições sobre como utilizar esses termos. Na prática podemos ter ontologias bastante complexas (com vários milhares de termos) ou ontologias bastante simples, comportando apenas um ou dois termos. As ontologias são as componentes básicas de construção para as técnicas de inferência na Web Semântica.

Na Web Semântica, a inferência pode ser interpretada como a descoberta de novas relações entre os conceitos. Os dados são modelados como um conjunto de nomes de relações entre recursos. A inferência significa que procedimentos automáticos podem gerar novas relações baseadas nos dados e em informação adicional na forma de um conjunto de regras. Se as novas relações são adicionadas ao conjunto dos dados, ou são retornadas em tempo real num questionário, é uma tarefa de implementação. Na Web Semântica, a fonte de tal informação extra pode ser definida através de vocabulários ou de conjuntos de regras. Ambas as aproximações recorrem a técnicas de representação do conhecimento. Em geral, as ontologias concentram-se em métodos de classificação, pondo ênfase na definição de "classes" e "subclasses", em como os recursos individuais podem ser associados a essas classes, e caracterizando as relações entre as classes e suas instâncias. A inferência na Web Semântica é uma das ferramentas de escolha para promover a qualidade da integração dos dados na web, através da descoberta de novas relações, analisando automaticamente os conteúdos dos dados, ou gerir o conhecimento na web em geral. As técnicas baseadas em inferência também são importantes na descoberta de possíveis inconsistências na integração dos dados.

O papel principal das ontologias na Web Semântica é ajudar e facilitar a integração dos dados quando, por exemplo, existem ambiguidades nos termos utilizados em conjuntos de dados diferentes, ou quando um pouco de conhecimento extra pode conduzir à descoberta de novas relações. Considere-se, por exemplo, a aplicação de ontologias na área de cuidados de saúde. Os profissionais médicos utilizam as ontologias para representar o conhecimento sobre sintomas, doenças e tratamentos. As companhias farmacêuticas utilizam as ontologias para representar a informação sobre medicamentos, dosagem e alergias. Combinar estes dois conhecimentos com os dados dos pacientes possibilita o desenvolvimento de toda uma gama de aplicações inteligentes, tais como: ferramentas de apoio à decisão que podem pesquisar sobre possíveis tratamentos; sistema que monitorizam a eficácia dos medicamentos e possíveis efeitos secundários; e ferramentas que suportam a investigação epidemiológica. Outro tipo de exemplo é utilizar as ontologias para organizar o conhecimento. Bibliotecas, museus, jornais, portais governamentais, empresas, aplicações de redes sociais, e outras comunidades que gerem grandes quantidades de coleções de livros, artefactos históricos, notícias, glossários de negócios, entradas de blogs, etc., podem agora utilizar os vocabulários, recorrendo a formalismos padrões, para promover o poder

dos dados interligados. O tamanho e a complexidade das ontologias e suas inferências dependem fortemente do propósito da sua criação e as aplicações decidirão se utilizam tudo ou apenas partes das representações.

A adaptação das lógicas descritivas como linguagens de ontologias tem sido destacada pelo seu importante papel como base na construção de várias linguagens web, incluindo a linguagem OWL, uma linguagem de ontologias normalizada para a Web Semântica desenvolvida pela W3C. A linguagem OWL tem uma sintaxe baseada em esquemas RDF, mas teve a sua origem na expressiva lógica descritiva $\mathcal{SHIQ}$ (lógica descritiva $\mathcal{ALC}$ estendida às restrições de cardinalidade, às qualidades transitivas e inversas, onde $\mathcal{S}$ é a abreviatura para a lógica descritiva $\mathcal{ALC}$ adicionada de qualidades transitivas).

A linguagem OWL é utilizada para definição e instanciação de ontologias web. Uma ontologia OWL pode formalizar um domínio, definindo classes e propriedades destas classes, definir indivíduos e afirmações sobre eles e, usando-se a semântica formal OWL, especificar como derivar consequências lógicas, i.e., factos que não estão presentes na ontologia, mas são vinculados pela semântica.

Definição 2.5.1 *Uma **Instância** é um objeto que corresponde à descrição lógica de um indivíduo.*

Definição 2.5.2 *Uma **Classe** é uma coleção de objetos que corresponde à descrição lógica de um conceito. Uma classe pode conter indivíduos, instâncias da classe. Uma classe pode conter qualquer número de instâncias. Uma instância pode pertencer a nenhuma, uma ou mais classes. Uma classe pode ser subclasse de outra classe, herdando as características da sua classe pai. Isto corresponde à subsunção lógica e inclusão de conceitos na lógica descritiva.*

Definição 2.5.3 *Uma **Propriedade** é uma relação binária que especifica as características de uma classe e que corresponde a uma regra da lógica descritiva. As propriedades são atributos das instâncias e algumas vezes funcionam como valores de dados ou ligações para outras instâncias. As propriedades podem possuir capacidades lógicas tais como transitivas, simétricas, inversas e funcionais. As propriedades podem conter domínios e contradomínios.*

As propriedades dos tipos de dados são relações que se estabelecem entre as instâncias de classes e literais RDF ou tipos de dados XML.

As propriedades dos objetos são relações que se estabelecem entre duas classes.

Definição 2.5.4 *A família de linguagens OWL suporta vários operadores sobre classes, tais como união, interseção e complemento. Aém disso, suporta também os operadores classes enumeráveis, cardinalidade de classes e disjunção de classes.*

A linguagem OWL possui três sub-linguagens[18] incrementais, projetadas para serem usadas por diferentes comunidades de implementadores e utilizadores:

- OWL Lite - é uma sub-linguagem da OWL DL que utiliza apenas algumas características da linguagem OWL e possui mais limitações do que OWL DL ou OWL Full;
- OWL DL - é uma linguagem com o máximo de expressividade: com completude (todas as conclusões são garantidas serem processadas pelos computadores) e decidibilidade computacional (todas os processamentos terminarão num tempo finito). Esta linguagem inclui todas as construções da linguagem OWL, podendo ser usadas sob certas restrições. A sigla DL possui correspondência com a lógica descritiva;
- OWL Full - é uma linguagem com o máximo de expressividade e independente sintaticamente da linguagem RDF. A OWL Full e a OWL DL suportam o mesmo conjunto de construtores da linguagem OWL, embora com restrições um pouco diferentes. Essas diferenças recaem em restrições sobre o uso de alguns desses construtores e no uso de recursos RDF. Enquanto a OWL Full permite misturar livremente as regras OWL com esquemas RDF e não requer uma separação estrita entre de classes, propriedades, indivíduos e valores de dados, i.e., por exemplo, uma classe pode ser simultaneamente uma classe e um indivíduo, a OWL DL coloca algumas condições na utilização de regras OWL conjuntamente com esquemas RDF e requer a disjunção entre classes, propriedades, indivíduos e valores de dados.

Cada uma destas sub-linguagens é uma extensão de sua antecessora, ou seja, cada ontologia válida em OWL Lite é uma ontologia válida em OWL DL, esta por sua vez é uma ontologia válida em OWL Full. Além disso, todo documento OWL (Lite, DL ou Full) é um documento RDF e todo documento RDF é um documento OWL Full. No entanto, apenas alguns documentos RDF são documentos OWL Lite ou OWL DL válidos.

Exemplo 2.5.1 Considere que relativamente ao conceito pizza[19] conseguimos estabelecer as seguintes afirmações:

```
Pizza has PizzaBase as its base.
Pizza is disjoint with PizzaBase.
NonVegetarianPizza is exactly Pizza that is not VegetarianPizza.
isIngredientOf is a transitive property.
isIngredientOf is inverse of hasIngredient.
```

A correspondente representação na sintaxe da lógica descritiva é:

[18] http://www.w3.org/TR/owl-features/

[19] Exemplo adaptado e retirado de http://www.obitko.com/tutorials/ontologies-semantic-web/owl-example-with-rdf-graph.html

$$\texttt{Pizza} \sqsubseteq \exists\, \texttt{hasBase.PizzaBase}$$
$$\texttt{Pizza} \sqcap \texttt{PizzaBase} \equiv \bot$$
$$\texttt{NonVegetarianPizza} \equiv \texttt{Pizza} \sqcap \neg\, \texttt{VegetarianPizza}$$
$$\texttt{Tr(isIngredientOf)}$$
$$\texttt{isIngredientOf} \equiv \textbf{inverse}\ \texttt{hasIngredient}$$

Cuja representação na sintaxe OWL utiliza as identificações URI para declarar todas as classes e propriedades:

```
Namespace(p = <http://example.com/pizzas.owl#>)
Ontology(<http://example.com/pizzas.owl#>
         Class(p:Pizza partial
               restriction(p:hasBase someValuesFrom(p:PizzaBase))
              )
         DisjointClasses(p:Pizza p:PizzaBase)
         Class(p:NonVegetarianPizza complete
               intersectionOf(p:Pizza complementOf(p:VegetarianPizza))
              )
         ObjectProperty(p:isIngredientOf Transitive
         inverseOf(p:hasIngredient))
        )
```

Quando se converte o exemplo expresso em OWL para a representação RDF, todas as declarações têm de ser transformadas para triplos:

```
@prefix :     <http://example.com/pizzas.owl#> .
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl:  <http://www.w3.org/2002/07/owl#> .

:Pizza rdfs:subClassOf
        [ a owl:Restriction ;
          owl:onProperty :hasBase ;
          owl:someValuesFrom :PizzaBase ] ;
       owl:disjointWith :PizzaBase .

:NonVegetarianPizza  owl:equivalentClass
        [  owl:intersectionOf
            (  [owl:complementOf :VegetarianPizza]
               :Pizza  )  ] .

:isIngredientOf
```

```
a owl:TransitiveProperty , owl:ObjectProperty ;
owl:inverseOf :hasIngredient .
```

▲

2.6 SPARQL, uma Linguagem de Consulta para Documentos RDF

Os *queries* (questionários ou consultas) na Web Semântica significam ter-se tecnologias e protocolos que permitam recuperar informação da web de dados, de forma programável. A família de especificações RDF, disponibilizada pela W3C, originalmente desenhada como um modelo de dados para metadados, tornou-se no método genérico para descrição ou modelação conceptual da informação que é implementada nos recursos web, utilizando uma variedade de notações de sintaxe e formatos de serialização de dados. A especificação RDF fornece as fundações para publicar e interligar os dados na web.

Atualmente a web de dados é tipicamente representada utilizando as especificações RDF como formato de dados e tem como linguagem para interrogar os dados a SPARQL (*SPARQL Protocol and RDF Query Language*), i.e., permite questionar os dados e obter os resultados através de *HyperText Markup Language* (HTML)[20] ou *Simple Object Access Protocol* (SOAP)[21]. A linguagem SPARQL permite consultar (questionar) documentos representados no formato RDF, ou seja, é uma linguagem para consultar e questionar bases de dados, capaz de devolver e manipular dados armazenados no formato RDF. A normalização da linguagem foi desenvolvida pelo grupo de trabalho RDF *Data Access Working Group* (DAWG)[22] pertencente à comunidade W3C, e é considerada uma das tecnologias chave para a Web Semântica. A linguagem SPARQL permite que um questionário seja constituído por padrões triplos, conjunções, disjunções e outros padrões opcionais[23], tais como filtros, ordenações, tipos de dados, etc. Tecnicamente, os *queries* (questionários) SPARQL são baseados em padrões descritos como triplos. Os padrões RDF podem ser vistos como um conjunto de relações entre recursos; os questionários SPARQL fornecem um ou mais padrões que vão de encontro a essas relações. Estes padrões triplos são semelhantes aos triplos RDF, exceto que um ou mais dos recursos constituintes são referências a variáveis. A ferramenta SPARQL retorna os recursos para todos os triplos que correspondem a esses padrões.

Utilizando as ferramentas SPARQL, da web de dados pode-se extrair informação possivelmente complexa (referências a recursos existentes e suas relações) que é devolvida, por exemplo, em formato de tabela. Esta tabela pode ser incorporada noutra página da web. Neste contexto, a tecnologia SPARQL fornece uma ferramenta poderosa para cons-

[20] http://www.w3.org/TR/xhtml1/
[21] http://www.w3.org/TR/2000/NOTE-SOAP-20000508/
[22] http://www.w3.org/2003/12/swa/dawg-charter
[23] http://www.w3.org/TR/rdf-sparql-query/

truir, por exemplo, páginas híbridas complexas ou motores de busca que incluem dados provenientes da Web Semântica.

A linguagem SPARQL especifica quatro variações de questionários para diferentes propósitos:

- Questionário SELECT - utilizado para extrair informações da base de dados e os resultados são devolvidos num formato de tabela;
- Questionário CONSTRUCT - utilizado para extrair informações da base de dados e transformar os resultados em triplos RDF válidos;
- Questionário ASK - utilizado para fornecer um resultado simples de Verdadeiro/Falso numa consulta às bases de dados;
- Questionário DESCRIBE - utilizado para extrair um gráfico RDF, i.e., um conjunto de declarações RDF da base de dados.

Cada uma destas formas de consulta contém um bloco inicializado por WHERE que permite condicionar a consulta, embora quando o questionário é do tipo DESCRIBE este bloco seja opcional.

Exemplo 2.6.1 Imagine que se pretende saber quais as capitais da Europa que têm menos de 200.000 habitantes. O questionário SPARQL que traduz esta consulta é:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpedia-owl <http://dbpedia.org/ontology/>
SELECT ?capitalInEurope ?country ?populationTotal
WHERE {
 ?capitalInEurope rdf:type <http://dbpedia.org/class/yago/CapitalsInEurope> .
 ?capitalInEurope dbpedia-owl:country ?country .
 ?country rdf:type dbpedia-owl:Country .
 ?capitalInEurope dbpedia-owl:populationTotal ?populationTotal .
FILTER (?populationTotal < 200000)
}
```

O questionário SELECT irá consultar a lista de todos os recursos que são capitais da Europa e filtra apenas os recursos que tenham um total de população inferior a 200.000. O resultado é uma tabela contendo os valores das três variáveis: ?capitalInEurope ?country ?populationTotal (PREFIX : http://dbpedia.org/resource/)

capitalInEurope	country	populationTotal
:Andorra_la_Vella	:Andorra	22256
:City_of_San_Marino	:San_Marino	4493
:Podgorica	:Montenegro	150977
:Valletta	:Malta	6966
:Sukhumi	:Abkhazia	39100
:Pristina	:Kosovo	198214
:Vaduz	:Liechtenstein	5342
:Reykjavík	:Iceland	119108

▲

2.7 DBpedia

O projeto DBpedia[24] [5] é o resultado do empenho de uma comunidade, em extrair informação estruturada a partir de Wikipédia[25] e tornar esta informação disponível na web. A DBpedia permite consultas sofisticadas sobre os dados da Wikipédia, bem como ligações com outros conjuntos de dados na web e conectados com os dados da Wikipédia. O trabalho desenvolvido tem como objetivo facilitar o uso da enorme quantidade de informação disponível na Wikipédia e inspirar novos mecanismos de navegação, ligações e melhorar a própria enciclopédia.

Como muitas outras aplicações web, a Wikipédia tem o problema de que seus recursos de procura fazem-se apenas através da pesquisa de texto completo, limitando consideravelmente o acesso a essa base de conhecimento. A Wikipédia também apresenta muitas das propriedades dos dados editadas colaborativamente: tem dados contraditórios, convenções taxonómicas incoerentes, erros, e até mesmo *spam*.

As bases de conhecimento desempenham um papel cada vez mais importante no reforço da inteligência da web e de pesquisa cooperativa e no apoio à integração de informação. Hoje, a maioria das bases de conhecimento abrangem apenas domínios específicos, são criados por grupos relativamente pequenos de engenheiros do conhecimento, e o custo de manter atualizados os domínios são bastante dispendiosos. Ao mesmo tempo, a Wikipédia tornou-se uma das fontes de conhecimento utilizadas pela maioria dos utilizadores da web e é mantida por milhares de colaboradores. O projeto DBpedia aproveita essa fonte gigantesca de conhecimento extraindo informações estruturadas da Wikipédia e tornando essas informações acessíveis na web sob os termos da *Creative Commons Attribution-ShareAlike 3.0 License* e *GNU Free Documentation License*.

A última versão do DBpedia é a 3.8, baseada em ligações que datam de finais de maio e início de julho de 2012.

[24] `http://dbpedia.org/About`
[25] `http://www.wikipedia.org/`

A versão inglesa da base de conhecimento DBpedia descreve atualmente mais de 3,77 milhões de coisas, dos quais 2,35 milhões são classificados numa Ontologia consistente, incluindo 764 mil pessoas, 573 mil lugares (incluindo 387 mil lugares populacionais), 333 mil trabalhos criativos (incluindo 112 mil álbuns de música, 72 mil filmes e 18 mil jogos de vídeo), 192 mil organizações (incluindo 45 mil companhias e 42 mil instituições educacionais), 202 mil espécies e 5.500 doenças.

Para além disso, a DBpedia fornece versões em 111 idiomas distintos. Todas estas versões juntas descrevem mais de 20,8 milhões de coisas, onde cerca de 10,5 milhões sobrepõem-se a (estão interligados com) conceitos da versão inglesa da DBpedia. O conjunto de dados completo da DBpedia caracteriza rótulos e resumos para 10,3 milhões de coisas distintas, em cerca de 111 diferentes idiomas; 8 milhões de ligações para imagens e 24,4 milhões de ligações HTML para páginas web externas; 27,2 milhões ligações de dados em conjuntos de dados RDF externos, 55,8 milhões de ligações a categorias Wikipédia e 8,2 milhões de categorias *Yet Another Great Ontology* (YAGO)[26] [105, 48]. O conjunto de dados consiste em 1,89 mil milhões de peças de informação (triplos RDF), onde cerca de 400 milhões são extraídos da edição inglesa da Wikipédia, 1,46 mil milhões são extraídos de outras edições nas restantes línguas, e cerca de 27 milhões são ligações a dados RDF externos.

A base de conhecimento DBpedia tem várias vantagens sobre bases de conhecimento existentes: cobre muitos domínios, representa um verdadeiro acordo de uma comunidade, que evolui automaticamente à medida que a Wikipédia muda, e é verdadeiramente multilingue. A base de conhecimento DBpedia permite uma vasta consulta à Wikipédia, por exemplo, "Name of all cities in Europe with more than 10.000 inhabitants" ou "Name of all Portuguese musicians from the 20th century". Ao todo, os casos de uso da base de conhecimento DBpedia são comuns e variam desde a gestão de conhecimento empresarial até à pesquisa web, de forma a revolucionar a pesquisa na Wikipédia.

2.7.1 A Base de Dados DBpedia

A base de dados DBpedia utiliza uma ampla ontologia de multi-domínio que derivou da Wikipédia. A versão inglesa da base de dados DBpedia descreve mais de 3,77 milhões de coisas com 400 milhões de factos.

A Wikipédia tornou-se uma das fontes de conhecimento utilizada por um crescente número de utilizadores da web e é mantida por milhares de colaboradores. Os artigos da Wikipédia são constituídos essencialmente por texto livre, mas também contêm diferentes tipos de informação estruturada, tais como: modelos de caixas de informação (*infoboxes templates*), informação categorizada, imagens, coordenadas geográficas, ligações para páginas web externas e ligações com edições de diferentes idiomas da Wikipédia.

A DBpedia utiliza a notação RDF como um modelo flexível de dados para representar a informação extraída e a publicar na web.

[26] `http://www.mpi-inf.mpg.de/yago-naga/yago/`

Cada um dos recursos descritos na base de dados DBpedia é identificado por uma referência URI da forma `http://dbpedia.org/resource/Name`, onde `Name` é retirado da URL cuja fonte é o artigo da Wikipédia, e que tem a forma `http://en.wikipedia.org/wiki/Name`.

Cada recurso DBpedia é caracterizado por um rótulo (*label*), um resumo curto e outro longo escritos em Inglês, uma ligação para a página correspondente da Wikipédia, e uma ligação para uma imagem representando o recurso (se disponível). Se um recurso existe em várias versões de diferentes idiomas da Wikipédia, então tanto o resumo curto, como o resumo longo, são escritos nesses idiomas e as ligações para as páginas dos diferentes idiomas na Wikipédia são adicionados à descrição. Assim, cada recurso da DBpedia está diretamente interligado com um artigo Wikipédia expresso na língua inglesa.

A DBpedia oferece três esquemas diferentes de classificação para os recursos:

1. As categorias da Wikipedia são representadas usando o vocabulário *Simple Knowledge Organization System* (SKOS)[27] e os termos *Dublin Core Metadata Initiative* (DCMI)[28].

2. A classificação YAGO [105, 48] é derivada do sistema de categorias da Wikipédia utilizando a WordNet. O projeto YAGO extrai apenas 14 tipos de relações, tais como `subClassOf`, `type`, `familyNameOf`, `locatedIn` de diferentes fontes de informação na Wikipédia. Uma das fontes é o sistema de categorias da Wikipédia (para `subClassOf`, `locatedIn`, `diedInYear`, `bornInYear`), e a outra são os redirecionamentos da Wikipédia. O projeto YAGO não representa a extração das caixas de informação tal como a DBpedia. Para determinar as relações `(sub)class`, o projeto YAGO não utiliza a hierarquia completa de categorias da Wikipédia, mas estabelece ligações de categorias com a hierarquia WordNet.

3. As ligações WordNet *Synset* (ver Secção 2.8) são geradas relacionando manualmente os modelos das caixas de informação dos artigos Wikipédia e os WordNet *synsets*, e adicionando a ligação correspondente a cada recurso que usa o modelo especifico. Em teoria, esta classificação deve ser mais precisa do que o sistema de categorias Wikipédia.

2.8 WordNet

A WordNet[29] [33] é uma vasta base de dados lexical da língua natural Inglês. Os nomes, os verbos, os adjetivos e os advérbios são agrupados em conjuntos de sinónimos cognitivos, denominados *synsets*, cada um expressando um conceito distinto. Os *synsets* estão interligados através dos significados conceptuais semânticos e das relações lexicais. A rede resultante de palavras e conceitos significativamente relacionados podem ser visualizados

[27] `http://www.w3.org/2004/02/skos/`
[28] `http://dublincore.org/documents/dcmi-terms/`
[29] `http://wordnet.princeton.edu/`

utilizando um navegador. A estrutura da base de dados WordNet torna-a numa ferramenta bastante útil para a linguística computacional e para o Processamento de Língua Natural.

A principal relação entre as palavras na WordNet é a sinonímia. Ou seja, relação de proximidade semântica entre duas ou mais palavras, que podem, por isso, ser usadas no mesmo contexto sem que haja alteração de significado do enunciado em que ocorrem. As palavras que verificam a relação de sinonímia entre si são denominadas de sinónimos. Os sinónimos são agrupados em conjuntos, denominados *synsets*. Cada um dos *synsets* é ligado com outro *synset* de acordo com um número pequeno de relações conceptuais. Adicionalmente, cada *synset* contém uma breve descrição (*gloss*) e, em muitos casos, uma ou mais pequenas frases ilustrando o uso dos membros *synset*. As palavras com vários significados distintos são representadas em vários *synsets* distintos, tornando cada par forma-significado único no WordNet.

Relativamente aos *synsets*, a relação de subordinação é a principal relação codificada que existe entre eles. Esta relação é também denominada hiperonímia, ou hiponímia ou relação ISA, e faz a ligação entre *synsets* mais genéricos, tais como relaciona "móvel" (*furniture*) com "peça de mobiliário" (*piece_of_furniture*), para os cada vez mais específicos, como "cama" (*bed*) e beliche (*bunkbed*). Assim, a base de dados WordNet afirma que a categoria "mobiliário" inclui "cama", que por sua vez inclui "beliche"; por outro lado, conceitos como "cama" e "beliche" compõem a categoria de "móveis". Todas as hierarquias de substantivos fundamentalmente vão até ao nó raiz "entidade" (*entity*). A relação hiponímia é transitiva: se uma "poltrona é uma espécie de cadeira", e se a "cadeira é um tipo de mobiliário", então podemos concluir que "uma poltrona é um tipo de mobiliário". A base de dados WordNet faz a distinção entre tipos (substantivos comuns) e instâncias específicas (pessoas, países e entidades geográficas). Assim, "a poltrona é um tipo de cadeira", "Barack Obama é um exemplo de um presidente". As instâncias são sempre nós terminais (folhas) na sua hierarquia.

Os verbos *synsets* são também dispostos em hierarquias. Os verbos colocados na parte inferior das árvores (*troponyms*) expressam formas cada vez mais precisas de caracterizar um evento, tal como em "comunicar - conversar - sussurrar" (*communicate-talk-whisper*). A forma específica expressa depende do campo semântico. Os verbos que descrevem eventos que necessariamente e unidirecionalmente implicam um ao outro estão ligadas, tais como "compra - pagamento" (*buy-pay*), "sucesso - tentar" (*succeed-try*), "mostrar - ver" (*show-see*), etc.

Os adjetivos *synsets* estão organizados em termos de antonímia. Pares de antónimos "diretos", como "molhado-seco" (*wet-dry*) e "novo-idoso" (*young-old*) refletem o forte contrato semântico entre os seus membros. Cada um desses adjetivos polares está ligado a uma série de adjetivos que lhe são semanticamente semelhantes: "seco" está ligado a "ressequido" (*parched*), a "árido" (*arid*) e a "desidratado" (*dessicated*); "molhado" (*wet*) está ligado a "encharcado" (*soggy*), "alagado" (*waterlogged*), etc. Os adjetivos semanticamente semelhantes são antónimos indiretos do membro de controlo do polo oposto, por exemplo o

adjetivo "ressequido" é antónimo indireto do adjetivo "molhado". Os adjetivos relacionais (*pertainyms*) apontam para os nomes de que são derivados, tal como em "criminoso-crime" (*criminal-crime*).

Relativamente aos advérbios, a base de dados WordNet inclui muito poucos (*hardly, mostly, really*, etc.), porque a maioria dos advérbios na língua natural inglesa derivam diretamente de adjetivos via afixação morfológica (surpreendentemente, estranhamente, etc).

A equipa de investigadores do projeto WordNet disponibiliza uma versão Prolog da base de dados WordNet (atualmente WordNet 3.0[30]), a qual é utilizada no nosso trabalho para a pesquisa de termos semelhantes semanticamente. Sarah Witzig, em [113], apresenta-nos uma documentação detalhada da base de dados Prolog da WordNet e um conjunto de predicados em Prolog que permitem utilizar/questionar a base de dados. A base de dados Prolog é particionada em diversos ficheiros, cujo nome identifica o tipo de relação estabelecido entre os *synsets*. Assim, as cláusulas Prolog, que armazenam informações básicas sobre palavras e seu grupo *synset*, são definidas no ficheiro de `wn_s.pl`. Cada cláusula contém uma palavra, um identificador ID *synset* atribuído, e alguma informação adicional. Existem outros 15 ficheiros `wn_operador.pl`, onde o `operador` corresponde às várias relações diferentes que se pode definir na WordNet. As relações, descritas pelos operadores, ou são de natureza semântica ou lexical. Existem nove tipos de relações semânticas definidas: hiponímia (`is-a`), e sua relação inversa hiperonímia; seis relações metonímicas (`part-of`) - `component-of`, `member-of`, `substance-of` e as suas relações inversas; e a relação de complemento (`complement-of`).

2.9 Conclusão

Neste capítulo apresentámos e relacionámos os conceitos "ontologias" e a "Web Semântica", evidenciando o facto de se terem tornado em metodologias fundamentais para representar o domínio conceptual do conhecimento, permitindo o aumento das capacidades semânticas na consulta de bases de dados e até na web. Para suporte ao desenvolvimento do nosso trabalho, apresentámos as linguagens de representação que permitem estabelecer regras automáticas para que os computadores possam aceder e interpretar os dados. Bem como, apresentámos as bases de dados utilizadas no presente trabalho. No próximo capítulo, apresentaremos os sistemas de Pergunta-Resposta, com ênfase na cooperação e estruturação de dados para a pesquisa na Web Semântica.

[30] `http://wordnet.princeton.edu/wordnet/download/current-version/`

Capítulo 3

Sistemas de Pergunta-Resposta

Neste capítulo apresentamos um resumo sobre os sistemas de Pergunta-Resposta, focando os trabalhos desenvolvidos que estão relacionados com sistemas cooperativos e têm como domínio de conhecimento a Web Semântica [90]. Em adição, introduzimos os conceitos "Processamento da Língua Natural" e "Estruturas de Representação do Conhecimento" inerentes ao nosso trabalho. O presente capítulo está estruturado da seguinte forma. Na Secção 3.1 introduzimos o conceito de sistemas de Pergunta-Resposta. Na Secção 3.2, apresentamos um conjunto de trabalhos desenvolvidos e propostas feitas no âmbito dos sistemas de Pergunta-Resposta, mais direcionados para a cooperação e para a Web Semântica. Na Secção 3.3, apresentamos a arquitetura típica de um sistema de Pergunta-Resposta. Na Secção 3.4, introduzimos algumas características sobre metodologias mais frequentemente utilizadas nos sistemas de Pergunta-Resposta. Na Secção 3.5, enumeramos um conjunto de desafios inerentes ao desenvolvimento de sistemas de Pergunta-Resposta. Na Secção 3.6, apresentamos alguns tópicos de pesquisa atuais. Na Secção 3.7, introduzimos o conceito de Processamento da Língua Natural e na Secção 3.8 abordamos o conceito de Estruturas de Representação do Conhecimento. Finalmente, na Secção 3.9, estabelecemos as conclusões.

3.1 Introdução

Os sistemas de Pergunta-Resposta procuram encontrar respostas concisas e objetivas a questões expressas em língua natural, colocadas pelos utilizadores na sua própria terminologia [46]. Estes sistemas são desenvolvidos na área das Ciências da Computação, estão

diretamente relacionados com o estudo da Recuperação de Informação [11, 79] e do Processamento da Língua Natural (*Natural Language Processing* (NLP), ver Secção 3.7) [53, 66], e dizem respeito à construção de sistemas que respondem de forma automática a questões colocadas pelos utilizadores em língua natural. De forma a melhorar a comunicação entre os sistemas e os seus utilizadores, as respostas também deverão ser expressas em língua natural e, quando apropriado, também devem ser informativas, completas e justificadas.

Para encontrar uma resposta a uma questão, um sistema de Pergunta-Resposta pode pesquisar tanto em bases de dados estruturadas, como num conjunto de documentos expressos em língua natural. O domínio de pesquisa pode variar de pequenos conjuntos de documentos armazenados localmente, a documentos organizacionais internos, a redes de relatórios compilados de notícias, e até mesmo a web. Os sistemas de Pergunta-Resposta têm como objetivo fornecer respostas objetivas às questões colocadas pelos utilizadores, através da consulta do seu domínio de pesquisa, a sua base de conhecimento.

A investigação nesta área lida com um vasto leque de tipos de questões [98], nomeadamente: factos, listas, definições, hipotéticas, semanticamente limitadas, e questões independentes da língua (*cross-lingual questions*). O conhecimento prévio do tipo de resposta esperada ajuda os sistemas de Pergunta-Resposta a extrair respostas corretas e objetivas a partir do conjunto de documentos, que compõem o seu domínio do conhecimento.

Os primeiros sistemas de Pergunta-Resposta foram desenvolvidos na década de 1960 e eram essencialmente interfaces em língua natural para sistemas inteligentes, construídos para domínios específicos. O avanço da web reintroduziu a necessidade de técnicas de pesquisa agradáveis ao utilizador e que reduzissem o excesso de informação, colocando novos desafios à investigação na automação de responder a questões.

A quantidade de informação na web aumentou de forma exponencial ao longo dos anos, com conteúdos que cobrem quase qualquer assunto. Como resultado, quando os utilizadores procuram determinada informação, ficam um pouco confusos com a vasta quantidade de informação devolvida pelos motores de busca. Virtualmente qualquer tipo de informação está disponível na web de uma forma ou de outra, rondando a ordem dos milhares de milhões de páginas web. Gerir tais quantidades de informação não é uma tarefa simples. Os motores de busca, tais como o Google e o Yahoo, retornam ligações, junto com fragmentos de texto, de todos os documentos como resposta ao pedido feito pelos utilizadores, e que permitirá que estes possam navegar pelos conteúdos, através de uma longa lista de resultados, para procurar a resposta pretendida.

O desenvolvimento dos sistemas de Pergunta-Resposta surgiu como uma tentativa de resolver este problema de sobrecarga de informação. De acordo com o domínio de conhecimento, os sistemas de Pergunta-Resposta podem classificar-se em duas categorias: os de domínio fechado e os de domínio aberto.

Os sistemas de Pergunta-Resposta de domínio fechado lidam com questões baseadas num domínio específico (por exemplo, medicina, música, carteira de clientes,

etc.). Podem ser vistos como sistemas mais simples, uma vez que os sistemas de Processamento da Língua Natural podem explorar domínios de conhecimento específicos, frequentemente formalizados em ontologias. Consistente com o papel das ontologias na estruturação e organização da informação semântica na web, os sistemas de pergunta-Resposta baseados em ontologias permitem explorar o poder expressivo das ontologias e ir para além das habituais consultas por palavras-chave correspondentes. O domínio específico de um sistema de Pergunta-Resposta envolve o uso intensivo de Processamento da Língua Natural, formalizado através da construção de uma ontologia do domínio considerado. Os domínios poderão referir-se a contextos onde um tipo limitado de questões são aceites.

Os sistemas de Pergunta-Resposta de domínio aberto lidam com questões sobre qualquer assunto, e só podem depender de ontologias genéricas e do conhecimento do mundo. Normalmente existe mais informação disponível de onde se pode extrair as respostas.

3.2 Uma Visão Geral sobre os Sistemas de Pergunta-Resposta

As mais importantes áreas de aplicação dos sistemas de Pergunta-Resposta são a extração de informação de toda a web, bases de dados disponíveis online e a pesquisa sobre sites individuais. Os atuais sistemas de Pergunta-Resposta [2] usam documentos de texto como recursos de conhecimento subjacentes e combinam várias técnicas de Processamento da Língua Natural (ver Secção 3.7) para pesquisar uma resposta. Com o objetivo de fornecer aos utilizadores respostas objetivas, os sistemas de Pergunta-Resposta necessitam de ir para além da análise sintática e lexical, passando pela análise semântica e o processamento de textos e recursos do conhecimento. Além disso, os sistemas de Pergunta-Resposta equipados com capacidades de raciocínio conseguem derivar respostas mais adequadas, através do recurso à representação semântica e a sistemas de raciocínio (tais como, Lógica Descritiva e Ontologias).

Uma abordagem sobre os sistemas de Pergunta-Resposta baseados em ontologias é apresentado em [73]. Um estudo sobre a utilização de interfaces em língua natural, e as expressões de pesquisa expressas em língua natural, direcionados para o utilizador, sobre domínios do conhecimento baseados em ontologias, é apresentado em [60]. Para este propósito, os autores introduzem quatro interfaces, cada um permitindo diferentes linguagens para as pesquisas e apresentam um estudo de utilização comparativa desses interfaces. Os resultados apresentados revelam uma clara preferência por expressões totalmente expressas em língua natural, e um conjunto limitado de expressões começadas com palavras-chave ou com estruturas formais.

Várias conferências e workshops têm-se focado em aspetos na área da pesquisa em sistemas de Pergunta-Resposta. Iniciado em 1999, a Conferência em Recuperação de Texto

(*Text REtrieval Conference* (TREC)[1]) tem financiado uma trajetória envolvendo os sistemas de Pergunta-Resposta e que tem por objetivo avaliar os sistemas que respondem a questões factuais, através da consulta dos documentos presentes no corpus da TREC. Um número significativo de sistemas nesta avaliação conseguiram combinar com sucesso técnicas de Recuperação de Informação e Processamento da Língua Natural. Em [4], os autores apresentam algumas revisões e comparam três principais abordagens de sistemas de Pergunta-Resposta baseadas em Processamento da Língua Natural, em Recuperação de Informação e em modelos de questões, evidenciando as principais diferenças e o contexto de aplicação que melhor se adequa a cada sistema.

3.2.1 Os Sistemas Cooperativos de Pergunta-Resposta

Os sistemas cooperativos de Pergunta-Resposta são sistemas de Pergunta-Resposta nos quais o sistema, tomando como partida uma questão inicial, tenta estabelecer um diálogo controlado com os utilizadores. Ou seja, o sistema colabora de forma automática com o utilizador, com o objetivo de obter e esclarecer determinada informação que lhe permita fornecer a resposta correta. Estes sistemas fornecem aos utilizadores informação adicional, respostas intermédias, respostas qualificadas, e/ou questões alternativas.

Uma forma de comportamento cooperativo é disponibilizar ao utilizador informação associada que, no processo de interpretação da questão e pesquisa da resposta, se considera relevante. A técnica de relaxar uma questão com o objetivo de capturar informação vizinha é um meio de se obter informação possível de ser relevante. Um sistema cooperativo de Pergunta-Resposta descrito em [35] usa técnicas de relaxação para identificar automaticamente novas questões que estão relacionadas com a questão inicial. Um estudo que consiste na adaptação de técnicas de aprendizagem, para máquinas, definidas no âmbito da extração de informação, aplicadas na extração de respostas em sistemas de Pergunta-Resposta, é apresentado em [52]. Os autores identificam as especificidades do sistema e apresentam os resultados obtidos no teste e comparação de três algoritmos, assumindo uma crescente abstração dos textos em língua natural.

Um formalismo para a representação semântica dedicado aos sistemas cooperativos de Pergunta-Resposta é apresentado em [16], o qual é baseado em estruturas conceptuais e lexicais, e representa de uma forma homogénea textos web, questões expressas em língua natural e as respostas relacionadas. O autor também apresenta e analisa os pré-requisitos necessários na construção de respostas cooperativas em função dos recursos, do conhecimento e do processo. Com o objetivo de valorizar os sistemas cooperativos de Pergunta-Resposta, o autor de [82] apresenta um conjunto de técnicas para melhorar os sistemas de Pergunta-Resposta e discute os potenciais e impactos da sua utilização.

Uma resposta cooperativa [28, 37] a uma questão é uma resposta indireta que é mais útil para o utilizador, do que uma resposta direta e literal pode ser. Uma resposta coo-

[1] http://trec.nist.gov/

perativa pode explicar a falha ocorrida na produção de resultados e/ou sugerir questões relacionadas para continuação da pesquisa. No caso em que são produzidos resultados, uma resposta cooperativa pode disponibilizar informação adicional, não explicitamente pedida pelo utilizador. As respostas cooperativas enquadram-se no contexto dos sistemas de Pergunta-Resposta de língua natural e foram originalmente motivadas pelo desejo de tornar o diálogo entre o sistema e o utilizador o mais próximo possível do diálogo entre humanos. De facto, o processamento de respostas cooperativas é preferível às técnicas usuais de extração de respostas com o foco nos utilizadores: primeiro, humaniza o sistema; segundo, permite a utilização de vocabulário adaptado; e por último, permite a introdução de informação que o utilizador não solicitou explicitamente, mas que poderá estar interessado.

Existem alguns exemplos de trabalhos que tentam construir respostas, em vez de apenas extrair e devolver. Em [31], os autores propõem um modelo para um sistema de Pergunta-Resposta, onde o sistema tenta, a partir da questão introduzida pelo utilizador, estabelecer um diálogo controlado com o utilizador. No diálogo, o sistema tem como objetivo identificar e sugerir ao utilizador novas questões, relacionadas com a inicial. O controlador do diálogo é baseado na estrutura dos conceitos armazenados na base de conhecimento, nas restrições do domínio, e nas regras de condicionamento específicas. Os autores em [38] apresentam um protótipo para um sistema que retorna respostas cooperativas, corrige conceitos em falta, tem a pretensão de ir ao encontro das necessidades dos utilizadores, e utiliza informação semântica sobre a base de dados para formular respostas coerentes e informativas. As principais características de estratégias lexicais, desenvolvidas pela capacidade intelectual dos humanos na tarefa de responder a questões, são apresentadas em [17]. O autor também mostra como estas estratégias podem ser reproduzidas na construção de sistemas de Pergunta-Resposta, em particular dos sistemas inteligentes cooperativos de Pergunta-Resposta. Um método de pesquisar respostas que estão na vizinhança da resposta à questão inicial, colocada pelo utilizador, e que pode ser utilizado para processar respostas, que servirão as necessidades e pretensões do utilizador, é apresentado em [36].

As técnicas de raciocínio avançadas utilizados nos sistemas de Pergunta-Resposta levantam novos desafios para os investigadores, uma vez que as respostas não são apenas extraídas diretamente dos textos ou das bases de dados estruturadas, mas também são construídas utilizando várias formas de raciocínio com o objetivo de gerar respostas explicadas e justificadas. A representação do conhecimento integrado e mecanismos de raciocínio permitem, por exemplo, responder antecipadamente a questões que poderão surgir e resolver situações onde a resposta não é encontrada na base de conhecimento. Estes sistemas devem identificar e explicar falsos pressupostos ou os vários tipos de conflitos encontrados na questão.

Farah Benamara apresenta vários trabalhos nesta área: um modelo baseado em lógica para gerar respostas intencionais e objetivas dentro de um sistema cooperativo de Pergunta-Resposta é proposto em [15]. Benamara desenvolveu várias categorias de formas intencionais e um cálculo intencional de profundidade variável, que permitem a geração de respostas intencionais com o melhor nível de abstração, e mostra que é possível gerar res-

postas naturais com um modelo base; em [14] apresenta uma abordagem para desenhar um sistema de Pergunta-Resposta baseado em lógica, WEBCOOP, que integra representação do conhecimento e técnicas de raciocínio avançadas para gerar respostas cooperativas a questões colocadas em língua natural na web. Este projeto foi desenvolvido num domínio relativamente limitado, que inclui uma série de aspetos de turismo (mais concretamente o transporte), requer o desenvolvimento de um extrator de conhecimento a partir de páginas web (semelhante a uma operação de extração do conhecimento em passagens resultantes de uma componente de recuperação de informação) e a elaboração de um robusto e preciso interprete de questões. As respostas fornecidas aos utilizadores são construídas no formato web, integrando técnicas de geração de linguagem natural com hipertextos, a fim de produzir respostas dinâmicas. As respostas em língua natural são produzidas utilizando formas semânticas construídas a partir de processos de raciocínio; um formalismo de representação semântica aplicada aos sistemas cooperativos pergunta e resposta é apresentado em [13], que é baseado em estruturas conceituais lexicais e representa homogeneamente textos web, questões expressas em língua natural, respostas relacionadas e respostas cooperativas.

3.2.2 Outros Sistemas de Pergunta-Resposta

START[2] [59] é um sistema de Pergunta-Resposta em língua natural que fornece aos utilizadores o acesso a informação multimédia através da utilização de anotações da língua natural (que são interpretadores de afirmações e de frases). O sistema START reformula as questões colocadas pelo utilizador em consultas Omnibase, estabelecendo a ligação entre a língua natural e a estrutura das bases de dados. Obmnibase [58] é uma base de dados virtual que fornece acesso aos recursos web. A base de dados Omnibase impõe o modelo triplo objeto-atributo-valor num conjunto restrito de recursos de dados web. Assim, o sistema START utiliza anotações da língua natural que descrevem o conteúdo de vários segmentos de informação. A consulta do utilizador é comparada com as anotações armazenadas na base do conhecimento e quando uma correspondência é encontrada, o segmento correspondente à anotação é devolvido ao utilizador como resposta.

Querix [61] é um sistema de Pergunta-Resposta baseado em ontologias, que depende de diálogos de clarificação no caso de ambiguidades. Este sistema é constituído por um interface ao utilizador, um gestor da ontologia, um analisador de questões, um centro de correspondência, um gerador de questões, uma componente do diálogo e uma camada de acesso à ontologia. As questões em língua natural são convertidas numa consulta SPARQL e utilizando o WordNet são identificados os sinónimos. O interpretador *Stanford* também é utilizado neste sistema para traduzir a questão em língua natural na correspondente árvore sintática. O sistema Querix não explora as técnicas semânticas baseadas em lógica.

PowerAqua [70, 71, 69] é um sistema de Pergunta-Resposta baseado em múltiplas ontologias que, tendo por base uma questão expressa em língua natural, é capaz de retornar respostas desenhadas de recursos relevantes distribuídos na Web Semântica. O sistema

[2] http://start.csail.mit.edu/

PowerAqua permite que o utilizador possa escolher a ontologia e colocar questões em língua natural que estejam relacionadas com o domínio coberto pela ontologia escolhida. A arquitetura do sistema e os métodos de raciocínio utilizados são independentes do domínio, dependendo da semântica da ontologia e usa recursos lexicais genéricos, tais como a WordNet [33]. O sistema é capaz de aprender a terminologia do utilizador, de forma a melhorar a sua experiência ao longo do tempo. O seu mecanismo de aprendizagem utiliza o raciocínio sobre a ontologia para aprender padrões mais genéricos, que poderão depois ser reutilizados para perguntas com contextos semelhantes. Neste sistema são utilizados dois modelos importantes: a componente linguística, que é utilizada na conversão das questões expressas em língua natural no correspondente formato triplo de consulta; e o serviço de relação de similaridade (*Relation Similarity Service*) que transforma o formato triplo de consulta num formato triplo de ontologia. O modelo de dados é baseado em triplos, do tipo Sujeito, Predicado, Objeto. O sistema PowerAqua evoluiu do anterior sistema denominado AquaLog [72, 74], um sistema de Pergunta-Resposta baseado em ontologias para intranets e limitado ao uso de apenas uma ontologia. O desempenho do AquaLog foi baseado nas medidas de precisão e recuperação e os tipos de falha são referidos separadamente. Em média 63,5% das respostas sucessivas são recuperadas a partir da ontologia com ambiente no domínio fechado.

PANTO [109] é um interface em língua natural portável para ontologias, que aceita o input em língua natural e o resultado é uma consulta SPARQL (ver Secção 2.6). O sistema adota um modelo de triplos de dados (RDF *triples*) para interpretar a árvore sintática da questão, obtida do interpretador *Stanford*[3] [62]. Depois o sistema PANTO mapeia os triplos de consulta para triplos na ontologia e que são representados com entidades da ontologia. Finalmente, conjuntamente com metas e modificadores extraídos das árvores sintáticas, os triplos na ontologia são interpretados como consultas SPARQL. O sistema PANTO foi avaliado com a base de dados geográfica Mooney, com 877 questões, e tendo como resultados 88,05% e 85,86% relativamente à precisão e à recuperação, respetivamente.

HITIQA[4] [104] é um sistema interativo de Pergunta-Resposta sobre domínio aberto e foi desenvolvido com o objetivo de ajudar analistas inteligentes, e outro tipo de utilizadores, a encontrar respostas relevantes para questões colocadas em língua natural, de forma eficiente, e que combina diálogos baseados no idioma e navegação visual no domínio da informação. O sistema utiliza técnicas baseadas em eventos e nos dados para o processamento semântico, bem como diálogos expressos em língua natural e é enriquecido com um interface de visualização avançada da informação, com o objetivo de devolver respostas objetivas às questões colocadas pelos analistas, acrescida de informação contextual relacionada.

FREyA [29, 30] é um interface interativo para questionar ontologias, que combina interpretação sintática com o conhecimento da ontologia para reduzir o esforço de perso-

[3]O interpretador *Stanford* é um analisador de língua natural que interpreta e constrói a estrutura gramatical das frases. Esta ferramenta é fornecida pela *Stanford Natural Language Processing Group* em `http://nlp.stanford.edu/software/lex-parser.shtml`.

[4]http://www.hitiqa.albany.edu/

nalização. As regras não são utilizadas neste sistema, em contrapartida o conhecimento codificado na ontologia é utilizado para interpretar e compreender a questão colocada pelo utilizador. A interpretação sintática é utilizada para obter uma resposta precisa. Neste modelo, os conceitos da ontologia são identificados e verificados inicialmente. Depois, uma consulta SPARQL é gerada e o tipo de resposta é identificado. A árvore sintática é gerada automaticamente utilizando o interpretador *Stanford*. O mapeamento da questão colocada pelo utilizador nos conceitos da ontologia é implementado de forma automática e, se necessário, utilizando a ajuda do utilizador. O modelo de classificação é utilizado para estabelecer uma ordenação adequada nos tipos de strings semelhantes. A resposta devolvida pelo sistema é no formato gráfico RDF. Os valores das medidas de precisão e recuperação, para os dados testados, atingiram o seu maior valor em 92,4%, considerando sempre os casos em que o sistema retorna respostas, mesmo que sejam parciais ou incorretas. Para medir a desempenho do sistema, foi implementado o algoritmo estatístico *Mean Reciprocal Rank* (MRR) definido para avaliar o processo de consultas. O valor do MRR é atingido aos 0,81, suportando uma elevada precisão e recuperação. O sistema FREyA evoluiu de um trabalho anterior QuestIO [106], um interface baseado em questões para ontologias que traduz uma questão, colocada em língua natural ou apenas baseada em termos numa consulta SPARQL, e retorna as respostas ao utilizador depois de executar a consulta à ontologia.

3.3 Arquitetura de um Sistema de Pergunta-Resposta

A arquitetura típica de um sistema de Pergunta-Resposta é constituída essencialmente por três fases principais e distintas: classificação da questão; recuperação de informação ou processamento dos documentos; e extração de informação.

A classificação das questões representa um papel essencial nos sistemas de Pergunta-Resposta, constitui a primeira fase e consiste em classificar as questões de acordo com um tipo definido, gerar o tipo de resposta esperada, extrair palavras-chave e reformular as questões em múltiplas questões semanticamente equivalentes. Reformular uma questão num conjunto de questões com significado semelhante também é conhecido por expansão da questão [26] e serve de base para aumentar e melhorar o desempenho do mecanismo de recuperação de informação.

A fase da recuperação da informação é muito importante para os sistemas de Pergunta-Resposta. Esta fase tem como principal objetivo recuperar informação (i.e., pesquisar por informação contida nos documentos) que possa ser útil ou relevante para responder à questão colocada pelo utilizador. Técnicas de recuperação de informação são aplicadas para recolher e extrair com sucesso respostas relevantes, no domínio do conhecimento. Se não for encontrada em qualquer documento uma resposta correta, a continuidade do processo na busca de uma resposta está terminada. A precisão e classificação dos fragmentos que são candidatos a resposta também podem afetar o desempenho do sistema, na fase de recuperação de informação.

Finalmente, o módulo da extração da resposta é um tópico em crescimento nos sistemas de Pergunta-Resposta, onde é feita essencialmente a classificação e validação das respostas candidatas. A extração da resposta estabelece a diferença entre o que é considerado um sistema de Pergunta-Resposta e o significado usual dado a um sistema de recuperação de texto. Esta fase consiste em, utilizando a informação obtida na fase da recuperação de informação, extrair a resposta correta à questão colocada pelo utilizador. A tecnologia de extração de resposta torna-se um fator influente e decisivo no sistema de Pergunta-Resposta para obter os resultados finais.

3.4 Metodologias Utilizadas

Os sistemas de Pergunta-Resposta são diretamente dependentes de uma boa pesquisa no corpus - sem os documentos que contenham a resposta, há muito pouco que os sistemas de Pergunta-Resposta podem fazer. Assim, faz sentido que conjuntos maiores de documentos possam proporcionar um melhor desempenho nos sistemas de Pergunta-Resposta, a menos que o domínio da questão seja ortogonal ao conjunto de documentos. A noção de redundância de dados em conjuntos massivos de documentos, tais como a web, i.e., pequenos excertos de informação que são suscetíveis de ser formulados de muitas maneiras diferentes, em diferentes contextos e documentos [67], conduz a dois benefícios: por ter a informação certa e aparecer em muitas formas, o peso feito nos sistemas de Pergunta-Resposta para realizar técnicas complexas de Processamento da Língua Natural, de forma a entender o texto, é menor; as respostas corretas podem ser filtradas de falsos positivos, tendo em conta que uma resposta correta pode aparecer mais vezes nos documentos do que respostas incorretas.

Os sistemas de Pergunta-Resposta, que utilizam documentos de texto em língua natural como domínio do conhecimento, apoiam-se nas técnicas de Processamento da Língua Natural tanto para o processamento da questão, como também para indexar ou processar o texto do corpus do qual as respostas são extraídas.

Um número crescente de sistemas de Pergunta-Resposta utilizam a web como seu corpus de texto e conhecimento. No entanto, muitas destas ferramentas (tais como Google, Yahoo) não produzem uma resposta agradável, informativa e cooperativa ao utilizador, que por sua vez empregam métodos superficiais (técnicas baseadas em correspondência entre palavras, modelos, etc.) para produzir uma lista de documentos contendo a resposta provável.

Nos atuais sistemas de Pergunta-Resposta [2], tipicamente, o classificador das questões determina o tipo de questão e o tipo de resposta esperada. Depois da questão ser analisada, o sistema normalmente utiliza vários módulos que aplicam técnicas de Processamento da Língua Natural cada vez mais complexas, numa quantidade progressivamente reduzida de texto. A recuperação de documentos utiliza motores de busca, para identificar os documentos ou parágrafos no conjunto dos documentos que são suscetíveis de conter a resposta correta. Posteriormente, um filtro seleciona pequenos fragmentos de texto que

contêm strings do mesmo tipo do que o da resposta esperada. Por exemplo, se a questão for "Who is the President of Portugal?", o filtro retorna o texto que contém nomes de pessoas. Finalmente, a extração da resposta procura por mais informações ou pistas no texto que determinam se a resposta candidata pode realmente responder à questão.

3.5 Desafios de Desenvolver Sistemas de Pergunta-Resposta

O desenvolvimento de sistemas de Pergunta-Resposta tem lançado vários desafios aos investigadores motivados, em grande parte, pelo aumento significativo da informação disponível, pelo avanço na tecnologia e pelas exigências e necessidades dos utilizadores. Pelo que, atualmente é possível enumerar um conjunto de problemas que continuam a ter toda a atenção no meio dos investigadores e que foram inicialmente identificados por um grupo de investigadores e apresentados em [24]:

Classes de questões - Diferentes tipos de questões requerem o uso de estratégias diferentes para encontrar a resposta. As classes de questões estão agrupadas hierarquicamente em taxonomias [98], permitindo que o seu tratamento seja mais eficaz.

Processamento das questões - A mesma informação pode ser expressa de várias formas. Um modelo semântico de interpretação e processamento de questões deverá reconhecer as questões que são equivalentes, independentemente da forma como são apresentadas. Este modelo deverá permitir a tradução de questões complexas numa séria de questões mais simples, deverá identificar ambiguidades e tratá-las no contexto ou através de clarificação interativa.

Contexto - As questões são normalmente colocadas num contexto e as respostas são fornecidas dentro desse específico contexto. O contexto pode ser utilizado para clarificar uma questão, resolver ambiguidades ou manter o controle de uma pesquisa realizada através de um conjunto de questões. Por exemplo, a questão "Why did Cavaco Silva visit Singapura in May 2012?" pode estar a perguntar porque foi para Singapura e não para outro país qualquer; porque foi em maio de 2012 e não antes ou depois; ou o que Cavaco Silva estava à espera de realizar com a sua visita. Se a questão é uma de uma série de perguntas relacionadas, as perguntas anteriores e as suas respostas podem orientar o sistema sobre as intenções de quem coloca a questão.

Recursos de dados - Antes de uma questão ser colocada, é necessário saber quais os recursos do conhecimento que estão disponíveis e que são relevantes. A veracidade e precisão das respostas poderá ser garantida por utilização de métodos de avaliação e plataformas de testes (*test-beds*). No entanto, o processamento da questão, assim como a avaliação da resposta é diretamente dependente dos dados contidos nos recursos. Se a resposta a uma questão não está presente no domínio do conhecimento, não interessa quanto o processamento da questão, a recuperação de informação e

a extração da resposta, sejam eficientes e otimizados, o resultado correto não será obtido.

Extração da resposta - A extração da resposta depende da complexidade da questão, do tipo de resposta fornecido pelo processamento da questão, dos dados atuais onde a resposta é procurada, dos métodos de pesquisa e do contexto da questão.

Formulação da resposta - Os resultados dos sistemas de Pergunta-Resposta deverão ser descritos o mais próximo possível da língua natural. Por exemplo, quando o classificador da questão indica que o tipo da resposta é um nome (pessoa, organização, etc.), uma quantidade (valor monetário, comprimento, tamanho, distância, etc.) ou uma data, a extração de um simples dado é suficiente. Para outros casos, a apresentação da resposta pode requerer o uso de técnicas de fusão que combinam as várias partes da resposta, obtidas de múltiplos documentos.

Respostas em tempo real - Existe a necessidade de desenvolver sistemas de Pergunta-Resposta capazes de extraírem respostas de conjuntos enormes de dados em alguns segundos, independentemente da complexidade da questão, do tamanho e da heterogeneidade dos recursos de dados ou da ambiguidade da questão.

Multilingue (*cross-lingual*) - A habilidade de responder a questões colocadas numa língua natural utilizando o corpus da resposta noutra língua natural (ou até várias), permite que os utilizadores possam consultar e usufruir de informação que não utilizam diretamente.

Interatividade/Cooperação - Por vezes a informação necessária pode não ser bem interpretada pelo sistema de Pergunta-Resposta, o processamento da questão pode falhar na classificação da questão, ou a informação necessária para a extração e geração da resposta não seja facilmente extraível. Nestes casos, o utilizador não só poderá querer reformular a questão, como também dialogar com o sistema no sentido de clarificar ou melhorar a interpretação da questão.

Raciocínio avançado - Cada vez mais os utilizadores esperam por respostas que estão fora do âmbito de textos escritos ou base de dados estruturados. Para melhorar os sistemas de Pergunta-Resposta com estas capacidades, é necessário integrar componentes de raciocínio que operem numa variedade de bases do conhecimento, i.e., mecanismos de raciocínio que codifiquem o mundo do conhecimento e do senso comum, assim como o conhecimento específico de uma variedade de domínios. A componente baseada em conhecimento dos sistemas de Pergunta-Resposta deve ser vista como um acréscimo, benefício para a melhoria do seu desempenho. Os principais benefícios são: personalização, como as respostas são recolhidas a partir de uma base de conhecimento, as respostas podem ser adaptadas ao utilizador; nível controlável de detalhe, de forma a que a resposta seja adequada ao nível de conhecimento do utilizador, o nível de detalhe pode ser controlado dinamicamente, através da quantidade de informação a ser apresentada na resposta e recolhida a partir da base de conhecimento; robustez, por inferir respostas em vez de extraí-las, os sistemas de

Pergunta-Resposta podem ter a capacidade de responder a questões não previstas e de resolver situações em que nenhuma resposta poderia ter sido encontrada nos recursos de informação.

Perfil do utilizador - O perfil do utilizador fornece informações sobre quem questiona, compreendendo o contexto dos dados, os esquemas de raciocínio utilizados frequentemente, as bases comuns estabelecidas entre os diferentes diálogos ocorridos entre o sistema e o utilizador. O perfil pode ser representado como um modelo predefinido, onde cada campo do modelo representa um recurso diferente do perfil.

Agrupamento de informação - O agrupamento de informação para os sistemas de Pergunta-Resposta é uma tendência que surgiu para aumentar a precisão dos sistemas de Pergunta-Resposta, através da redução do espaço de pesquisa [97].

3.6 Tópicos de Pesquisa Atuais

Nos últimos anos, os sistemas de Pergunta-Resposta evoluíram no sentido de incorporarem domínios adicionais do conhecimento [81, 6]. Por exemplo, os sistemas de Pergunta-Resposta têm vindo a ser desenvolvidos para responder automaticamente a questões de contexto temporal e geoespacial, questões de definições e terminologias, questões biográficas, questões multilingue, e questões sobre conteúdos áudio, imagens ou mesmo vídeo. Os tópicos de pesquisa atuais, no âmbito dos sistemas de Pergunta-Resposta incluem:

Interatividade, cooperação e clarificação de questões e/ou respostas - a promoção e o desenvolvimento de sistemas que dialoguem com os seus utilizadores, com o objetivo de clarificar ambiguidades nas questões, personalizar e justificar as respostas, tornam os sistemas mais eficazes, eficientes e melhoram o seu desempenho. Uma visão geral sobre os sistemas interativos de Pergunta-Resposta e sua contextualização no campo mais vasto dos sistemas de Pergunta-Resposta é apresentada em [110]. Os autores estabelecem ainda as ligações existentes com a investigação em Recuperação de Informação e os sistemas cooperativos.

Reutilização das respostas - um sistema avançado de Pergunta-Resposta, com controlo de qualidade, deve ser capaz de acumular perguntas, respostas e informações auxiliares e reutilizá-las de forma a melhorar o processo de responder a perguntas futuras [81].

Representação do conhecimento e raciocínio - a interpretação correta de uma questão expressa em língua natural, a extração do conhecimento e informação necessários do domínio de conhecimento, e a integração de estratégias de raciocínio permitem aos sistemas de Pergunta-Resposta terem um desempenho mais eficaz aquando da construção da resposta a devolver ao seu utilizador [102].

Análise dos mídia social e os sistemas de Pergunta-Resposta - o elevado crescimento de conteúdos de mídia social (fotos, vídeos, marcações, ligações, etc.) e de

serviços associados, o apoio explícito para as interações sociais entre os utilizadores, como publicação de comentários, partilha de conteúdos, responder a questões e comentários tornam as redes sociais únicas. Neste contexto, o conteúdo utilizado e gerado pelos utilizadores é diferente, do conteúdo tradicional disponibilizado na web, em estilo, em qualidade, em autoria e em apoio explícito para as redes sociais. Pelo que, as redes sociais requerem novas técnicas de análise e recuperação de conteúdos relevantes [45, 44, 18, 1].

Análise de sentimento (mineração de opinião) - a crescente utilização de redes sociais e blogs, onde os utilizadores expressam as suas opiniões sobre os mais variados assuntos, aumentou substancialmente o conjunto de documentos disponibilizados na web. Paralelamente, o interesse na procura de opiniões escritas nesses documentos também aumentou, requerendo novas técnicas de análise e recuperação de conteúdos que permitam identificar manifestações de sentimento (positivo, negativo ou neutro) sobre uma entidade, por exemplo uma pessoa, um produto ou uma instituição, quando se procura uma determinada opinião em texto [95, 68, 64].

3.7 Processamento da Língua Natural

O Processamento da Língua Natural [53, 66] é uma subárea da Inteligência Artificial e da Linguística que surgiu na década de 1950 e tem como foco principal o estudo dos problemas da geração e compreensão automática de línguas naturais. Entende-se por língua natural qualquer linguagem que surge de forma não premeditada, como resultado da facilidade para a linguagem possuída pelo intelecto humano. A língua natural é normalmente utilizado para a comunicação, pode ser falada, assinada, ou escrita, e diferencia-se claramente de línguas construídas e linguagens formais, tais como programação de computadores, ou as "linguagens" usadas no estudo da lógica formal, especialmente a lógica matemática. Os sistemas de geração de língua natural convertem informação de bases de dados de computadores em linguagem normalmente compreensível ao ser humano, e sistemas de compreensão de língua natural convertem ocorrências da linguagem humana em representações mais formais, mais facilmente manipuláveis por programas de computador.

A comunicação entre as pessoas através da linguagem acompanhou a evolução da humanidade e das sociedades. Atualmente essas linguagens caracterizam-se por serem bastante estruturadas e complexas. Os humanos foram ganhando a capacidade natural de estabelecer um diálogo utilizando essas linguagens, i.e., têm a capacidade de interpretar a informação transmitida, raciocinar sobre ela, tirar conclusões e responder com nova informação. Atualmente os computadores e equipamentos eletrónicos obrigam-nos a aprender formas não intuitivas de comunicação com essas máquinas através de comandos precisos, linguagens de programação, menus, ligações e botões. Os interfaces entre as máquinas e seres humanos estão a ficar mais sofisticados e a caminhar aos poucos em direção às formas mais humanas de comunicação.

O Processamento da Língua Natural é o conjunto de métodos formais para analisar textos e gerar frases escritas num idioma humano. Normalmente os computadores estão aptos a compreender instruções escritas em linguagens de programação como o Java, C, Prolog, etc., mas possuem muita dificuldade em entender comandos escritos numa linguagem humana. Isso deve-se ao facto das linguagens de programação serem extremamente precisas, contendo regras fixas e estruturas lógicas bem definidas, que permitem ao computador saber exatamente como deve proceder a cada comando. No entanto, numa língua natural uma simples frase normalmente contém ambiguidades e interpretações que dependem do contexto, do conhecimento do mundo, de regras gramaticais, culturais e de conceitos abstratos.

O objetivo final do Processamento da Língua Natural é fornecer aos computadores a capacidade de entender e compor textos, ou seja, que o computador seja capaz de reconhecer o contexto, fazer análise sintática, semântica, léxica e morfológica, criar resumos, extrair informação, interpretar os sentidos e até aprender conceitos com os textos processados.

O Processamento da Língua Natural é estabelecido através dos seguintes cinco passos:

Análise Morfológica e Lexical corresponde ao primeiro nível do Processamento da Língua Natural e consiste em enquadrar as palavras das frases no contexto morfológico e lexical que caracterizam a linguagem. O léxico de uma linguagem é o seu vocabulário, que inclui as suas palavras e expressões. A morfologia é a identificação, análise e descrição da estrutura das palavras. As palavras são geralmente aceites como sendo as mais pequenas unidades da sintaxe. A sintaxe refere-se ao estudo das regras e princípios que regem a estrutura de frase de qualquer linguagem natural. A análise lexical consiste em dividir o texto em língua natural em parágrafos, frases e palavras e não pode ser feita separadamente da análise morfológica e sintática.

Análise Sintática consiste em analisar as palavras na frase de forma a obter a estrutura gramatical da frase. As palavras são transformadas em estruturas que mostram como as palavras estão relacionadas umas com as outras. Algumas sequências de palavras podem ser rejeitadas, se não respeitarem as regras da linguagem que determinam como as palavras podem ser combinadas.

Análise Semântica determina os possíveis significados das frases no seu contexto. As estruturas criadas pela análise sintática são enriquecidas com um significado, i.e., é feito um mapeamento entre as estruturas sintáticas e os objetos no domínio considerado. As estruturas para as quais não existe mapeamento são rejeitadas.

Integração do Discurso consiste em enquadrar o significado de cada frase no contexto global. O significado individual de cada frase pode depender das frases que as precedem, como também podem influenciar o significado das frases que as seguem.

Análise Pragmática traduz-se na interpretação das frases de forma a compreender o uso intencional da linguagem em determinadas situações, particularmente no que

concerne aos aspetos da linguagem que requerem o conhecimento do mundo, i.e., o que foi dito é reinterpretado de forma a determinar o que realmente significa.

Ao longo das últimas seis décadas, as principais áreas de aplicação do Processamento da Língua Natural são: compreensão da língua natural (*Natural Language Understanding*), geração da língua natural (*Natural Language Generation*), reconhecimento do discurso e da voz (*Speech or Voice recognition*), tradução automática (*Machine Translation*), correção da ortografia (*Spelling Correction*) e verificação da gramática (*Grammar Checking*).

3.8 Estruturas de Representação do Conhecimento

A Teoria da Representação do Discurso (*Discourse Representation Theory*[5]) é o nome específico do trabalho realizado por Hans Kamp [56, 54, 55, 57] no âmbito da interpretação dinâmica da língua natural, i.e., sobre a semântica da língua natural. O principal objetivo é associar frases expressas em língua natural a expressões lógicas que representem o seu verdadeiro significado. Esta teoria foi tornou-se gradualmente num termo genérico para propostas que permitam a interpretação dinâmica de línguas naturais. Estas propostas têm em comum o facto de que cada nova frase é interpretada em termos da sua contribuição, para uma já existente interpretação do discurso. As condições de interpretação para as frases são dadas como instruções para atualizar a representação do discurso.

A ideia básica da proposta é que um discurso em língua natural (uma sequência de frases proferidas pelo mesmo orador) é interpretada no contexto de uma estrutura de representação.

Definição 3.8.1 *O resultado do processamento de uma parte do discurso no contexto de uma estrutura de representação R é uma nova estrutura de representação R'; esta nova estrutura R' pode ser vista como uma atualização da versão R.*

A ideia base da teoria da representação do conhecimento consiste em:

- Não ser baseada em frases. A Teoria da Representação do Discurso constrói representações do discurso e não de frases. Estas representações são denominadas de Estruturas de Representação do Discurso (*Discourse Representation Structures* (DRS)). Para além de se construir uma DRS para cada frase, constrói-se uma DRS para cada frase de acordo com a sua contribuição no contexto das representações das restantes frases do discurso.
- Não está intimamente ligada à sintaxe de uma língua natural, nem a uma teoria particular da sintaxe. Qualquer análise sintática capaz de determinar o significado pode ser usado numa implementação da DRS.

[5] `http://plato.stanford.edu/entries/discourse-representation-theory/`

- Restrições estruturais sobre a acessibilidade dos antecedentes anafóricos são previstos corretamente.
- Uma teoria de condições de verdade é construída com a teoria da representação do discurso. A verdade é definida em termos de incorporação da DRS num modelo. Assim, a teoria da representação do discurso preserva as vantagens da lógica de predicados como uma linguagem de representação, ao trazer a formalização mais perto da estrutura de linguagem natural.

Neste contexto, a definição genérica da estrutura de representação do discurso é estabelecida da seguinte forma.

Definição 3.8.2 *Uma DRS é um conjunto de referentes, variáveis universalmente quantificadas e um conjunto de condições (Predicados de Primeira-Ordem). As condições podem ser ou atómicas (do tipo $P(u_1, ..., u_n)$ ou $u_1 = u_2$) ou complexas (negação, implicação, disjunção, conjunção ou quantificadores generalizados).*

Mais concretamente, podemos estabelecer que a Estrutura de Representação do Discurso, a DRS, é um par ordenado $< U, Con >$, onde U é o universo do discurso, i.e., o conjunto das entidades do discurso, e Con o conjunto de condições, i.e., predicados ou igualdades que estas entidades devem verificar.

Exemplo 3.8.1 Considere, por exemplo, a seguinte frase

`Maria owns a dog.` (3.1)

é representada pela DRS

U = X,Y
Con = name(X,“Maria”), dog(Y), owns(X,Y) (3.2)

ou no formato mais usual (formato de caixas):

X, Y
name(X,“Maria”) dog(Y) owns(X,Y)

▲

A construção e veracidade de uma DRS é definida através da sua integração num modelo. Um modelo inclui um domínio D, i.e., um conjunto de entidades que podem ser discutidas, e uma função de interpretação I que faz o mapeamento de todos os predicados n-ésimos contidos no discurso no conjunto de n-tuplos de elementos de D, e mapeia todas as constantes lógicas da linguagem em indivíduos de D. Intuitivamente, D é o conjunto de coisas de que se pode falar (incluindo entidades imaginárias e reais).

3.9 Conclusão

O principal objetivo dos sistemas de Pergunta-Resposta é devolver respostas objetivas a questões colocadas pelos utilizadores, em vez de devolver listas de documentos completos ou fragmentos que melhor se aproximam da resposta pretendida, como a maioria dos sistemas de Recuperação de Informação. Neste capítulo apresentámos um resumo sobre os sistema de Pergunta-Resposta, focando os trabalhos desenvolvidos que estão relacionados com sistemas cooperativos e têm como domínio de conhecimento a Web Semântica, evidenciando aspetos da arquitetura típica de um sistema de Pergunta-Resposta, algumas características sobre metodologias utilizadas mais frequentemente nos sistemas de Pergunta-Resposta, os desafios inerentes ao desenvolvimento de sistemas de Pergunta-Resposta e alguns tópicos de pesquisa atuais.

O vasto leque de desafios associados ao desenvolvimento de sistemas de Pergunta-Resposta; a crescente necessidade de motores de busca inteligentes capazes de satisfazer as exigências dos mais variados tipos de utilizadores da web; a necessidade de interação e cooperação entre os utilizadores e os sistemas; a necessidade de apresentar, por parte do sistema, respostas objetivas, informativas e expressas o mais próximo da língua natural; foram motivações suficientes para tomarmos a decisão de avançar com a árdua tarefa: a construção de um sistema cooperativo de Pergunta-Resposta para a Web Semântica; que será apresentado detalhadamente nos próximos capítulos.

Capítulo 4

Ferramenta Cooperativa de Pergunta-Resposta para OWL

Neste capítulo apresentamos um sistema cooperativo de Pergunta-Resposta que recebe questões expressas em língua natural e tem a capacidade de retornar uma resposta cooperativa, também expressa em língua natural, obtida dos recursos na Web Semântica (ontologias e descrições OWL2). O sistema inicia um diálogo controlado sempre que existir alguma ambiguidade na questão ou quando deteta que está na presença de múltiplas respostas. Este capítulo está organizado da seguinte forma. Primeiro, na Secção 4.1, fazemos uma introdução do sistema proposto. Nas secções seguintes descrevemos as componentes mais importantes da sua arquitetura, nomeadamente: na Secção 4.2, a componente Análise Sintática; na Secção 4.3, a componente Interpretação Semântica; na Secção 4.4, a componente Contexto Ontológico; na Secção 4.5, a componente Avaliação Semântica; e na Secção 4.6, a principal componente do sistema, o Controlador do Discurso. Em paralelo, expomos um exemplo ilustrativo das funcionalidades do sistema. Finalmente, na Secção 4.7, apresentamos algumas ideias e conclusões a retirar deste capítulo.

4.1 Introdução

Os sistemas de Pergunta-Resposta para as línguas naturais na Web Semântica requerem mecanismos capazes de interpretar para além da correspondência entre palavras nos documentos, atualmente feito pelos mecanismos de pesquisa na web, e de encontrar a resposta objetiva, sem requerer a ajuda dos utilizadores para interpretar os documentos retornados.

Sem quaisquer mecanismos de acesso e comunicação, é difícil para a generalidade dos utilizadores finais compreenderem a complexidade da Web Semântica, baseada em lógica. Torna-se crucial permitir que um utilizador comum da web beneficie o máximo da poderosa expressividade dos modelos de dados da Web Semântica, enquanto se esconde a sua complexidade. A disponibilidade de interfaces amigáveis, que permitem utilizar as vastas redes de dados na web e que apoiem e suportem os utilizadores finais na pesquisa e consulta destas fontes de informação heterogéneas, é uma necessidade emergente.

Consistente com a função desempenhada pelas ontologias na estruturação da informação semântica na web, os sistemas de Pergunta-Resposta baseados em ontologias permitem explorar a poderosa expressividade das ontologias e ir para além das usuais pesquisas baseadas em palavras.

O sistema cooperativo de Pergunta-Resposta para ontologias OWL [89, 87, 88, 85, 86] que apresentamos recebe questões, expressas em língua natural (atualmente utilizamos apenas o Inglês), e tem a capacidade de retornar uma resposta cooperativa, também expressa em língua natural, baseada nos recursos da Web Semântica, mais especificamente, na DBpedia [5], representada em OWL/RDF, como base de conhecimento e a WordNet [33] para construir questões semelhantes. O sistema inicia um diálogo controlado sempre que existir alguma ambiguidade na questão ou quando deteta que está na presença de múltiplas respostas. O sistema inclui análise em profundidade (*deep parsing*[1]), o uso de ontologias, repositórios lexicais e semânticos, tais como a WordNet, e recursos web, tais como a DBpedia.

Os recursos do sistema baseiam-se em ontologias, não apenas para a interpretação da informação, mas também para encontrar as respostas às questões colocadas. O nosso objetivo é proporcionar um sistema que seja independente do conhecimento prévio dos recursos semânticos por parte do utilizador e que seja capaz de responder cooperativamente às questões colocadas pelo utilizador na sua terminologia, a língua natural.

O sistema tem a capacidade de clarificar os problemas de ambiguidade, recorrendo a um diálogo controlado com o utilizador, necessário para encontrar o caminho para a resposta correta. Se a resposta a uma questão for múltipla, e não gerar questões de ambiguidade, as soluções serão agrupadas de acordo com o seu contexto semântico, fornecendo uma resposta cooperativa, informativa e clara ao utilizador.

O sistema mantém a estrutura do diálogo, que fornecerá o contexto para a interpretação da questão, que inclui: contextos implícitos, tais como conhecimento temporal e espacial; entidades e informação útil para a interpretação semântica, tais como entidades do discurso usadas para a resolução de anáforas, na procura do verdadeiro significado de uma instância de uma expressão.

[1] A análise em profundidade está diretamente relacionada com as propriedades da gramática. Consiste em, para uma dada frase, construir todos os possíveis subconjuntos dos elementos justapostos que podem descrever a categoria sintática. Um subconjunto é positivamente caracterizado se satisfaz as restrições da gramática.

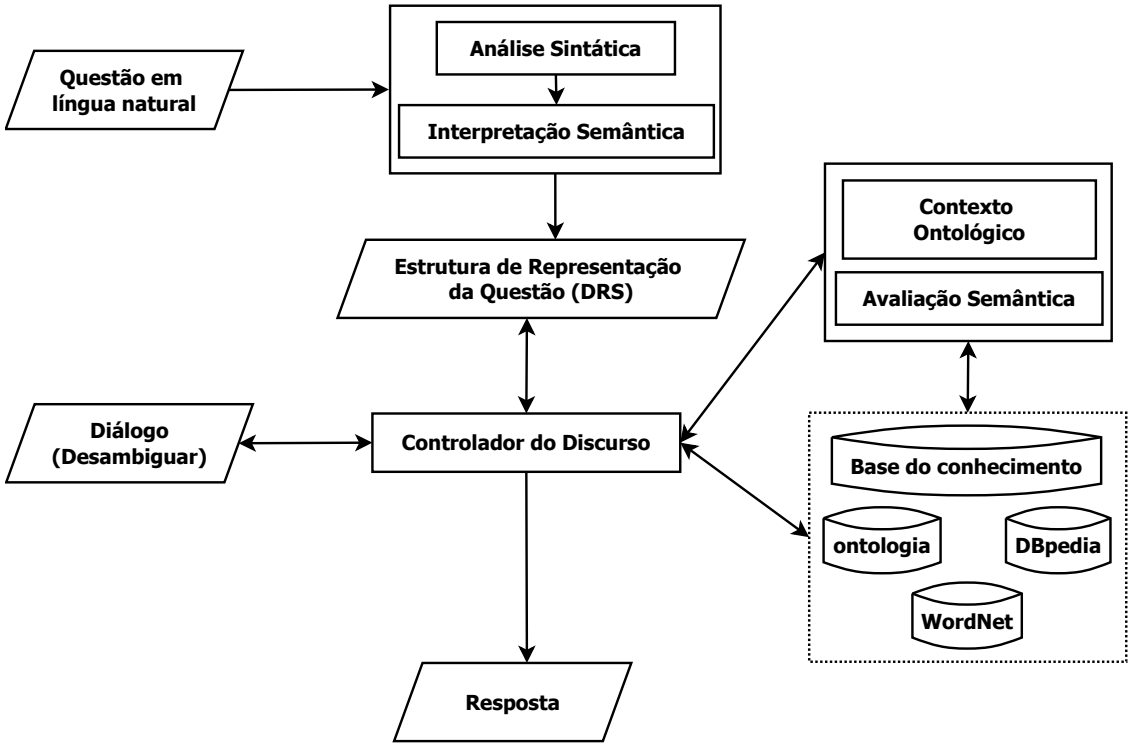


Figura 4.1: Arquitetura do Sistema Cooperativo de Pergunta-Resposta para a Web Semântica.

A arquitetura do sistema é apresentada na Figura 4.1 e, para ajudar a perceber o seu funcionamento, nas próximas secções descrevemos as suas componentes principais. Resumidamente, o sistema recebe a questão em língua natural e transforma-a na sua representação semântica, utilizando Estruturas de Representação do Discurso (DRS), ver Secção 3.8. Após a consulta dos recursos semânticos na base de conhecimento, fornece ao utilizador uma resposta em língua natural.

Os sistemas de Pergunta-Resposta dependem fortemente do raciocínio, facto que nos levou a escolher a Programação Lógica, mais especificamente a linguagem Prolog, para o seu desenvolvimento. Para além disso, existe uma vasta quantidade de bibliotecas e extensões do Prolog que permitem processar e questionar as ontologias OWL2, assim como incorporar a noção de contexto no processo de raciocínio. Em adição, o WordNet disponibiliza uma exportação para Prolog, facilitando a sua utilização. Como complemento, Wielemaker [111] fornece um estudo para a tradução e otimização de consultas, mais especificamente o *SeRQL RDF query language*, onde as consultas são traduzidas para factos Prolog, otimizados pela ordenação de literais. Finalmente, em [112] os autores apresentam uma forma de como desenvolver uma aplicação para a Web Semântica completamente em Prolog.

4.2 Análise Sintática

A Análise Sintática é feita com base na estrutura gramatical da questão. As palavras que compõem a questão são transformadas em estruturas que mostram como as palavras estão relacionadas entre si.

O sistema recebe a questão expressa em língua natural e, através da Análise Sintática, é gerada a correspondente árvore de derivação, obtida da interpretação gramatical. Para obter a estrutura sintática da questão, utilizamos a ferramenta fornecida pelo *The Stanford Natural Language Processing Group*[2] [62] - *Stanford parser*, um analisador estatístico.

O analisador *Stanford* utiliza o conjunto de marcações *Penn Treebank* [80] como marcador para a língua natural Inglesa. Na Tabela 4.1 estão descritas algumas dessas marcações e que serão utilizadas nos exemplos apresentados.

A Análise Sintática não impõe fortes restrições sobre a estrutura da questão e deixa todas as possíveis análises de interpretação e filtragem para o analisador semântico. A Análise Sintática transforma então a questão no correspondente conjunto de estruturas sintáticas.

Exemplo 4.2.1 De forma a ilustrar a execução do sistema de Pergunta-Resposta, considere a questão

```
Have all French romantic writers died?
```
(4.1)

Por aplicação do analisador *Stanford* obtemos a estrutura sintática[3]

```
(ROOT
  (SQ (VBP Have)
    (NP (DT all)
      (ADJP (NNP French) (JJ romantic))
      (NNS writers))
    (VP (VBN died))
    (. ?)))
```

▲

A estrutura sintática da questão será utilizada pelo analisador semântico, de forma a obter a interpretação e representação semântica da questão.

[2] `http://nlp.stanford.edu/software/lex-parser.shtml`

[3] No contexto da questão o termo "French" não é um nome próprio singular (NNP) mas um adjetivo (JJ). O termo "French" qualifica os "romantic writers", pelo que o substantivo plural (NNS) não é apenas o termo "writers" mas os dois termos "romantic writters". O analisador Stanford qualificou o termo "French" como sendo um nome pelo facto de estar escrita com letra maiúscula. Desta forma, a análise feita pelo Standford parser não é a mais correta. No entanto, a análise semântica irá ser feita utilizando sempre a estrutura sintática obtida por aplicação do Stanford parser.

Marcação	Descrição
ADJP	Adjetivo que compõe a frase/questão
DT	Determinante
JJ	Adjetivo
NP	Frase/Grupo nominal
NNP	Nome próprio, singular
NNS	Substantivo, plural
ROOT	Raiz da estrutura sintática da frase/questão
S	Expressão declarativa simples
SBAR	Expressão introduzida pela conjunção subordinada
SBARQ	Questão direta introduzida por "*wh-word*" ou "*wh-phrase*", i.e., questões ou frases que contenham palavras inicializadas por "wh" (*who, which, what, why, ...*)
SQ	Questão invertida do tipo "yes/no" ou expressão principal de uma questão do tipo "wh", seguida de uma frase "wh" em SBARQ
VBP	Verbo, presente singular mas não na terceira pessoa
VBN	Verbo, particípio passado
VBZ	Verbo, terceira pessoa no presente do singular
VP	Frase/Grupo verbal
WHADVP	Questão/Frase/Grupo de Advérbio "wh" indireta
WHAVP	Frase/Grupo de Advérbio "wh"
WHNP	Frase substantiva "wh"
WP	Pronome "wh"
WRB	Advérbio "wh"

Tabela 4.1: Descrição de algumas das marcações do discurso Penn Treebank para a língua Inglesa.

4.3 Interpretação Semântica

A Interpretação (Análise) Semântica é feita com base na Lógica de Primeira Ordem [47], estendida com quantificadores generalizados [12]. Existe um cuidado especial no tratamento e identificação das entidades do discurso, a fim de se ter o quantificador apropriado introduzido pela interpretação sintática. Nesta fase, a estrutura sintática da questão é reescrita na correspondente DRS, suportada pela Teoria da Representação do Discurso [56, 20].

A implementação desta componente segue uma abordagem semelhante à proposta em [99], utilizada para a construção de um sistema de Pergunta-Resposta sobre bases de dados de documentos. Este sistema é constituído por dois módulos separados: primeiro, é feita a análise dos documentos (extração de informação); e depois o processamento das questões

(recuperação de informação). O sistema procura fazer o processamento do corpus e da questão, suportado por teorias da linguística computacional: análise sintática (restrições gramaticais), usando a análise em profundidade; seguida de análise semântica, usando a Teoria da Representação do Discurso; e finalmente a interpretação semântica (pragmática), usando ontologias e inferência lógica.

A técnica utilizada nesta componente consiste, simplesmente, em identificar algumas marcações na estrutura sintática da questão inicial e reescrevê-las com um significado semântico, de acordo com algumas das seguintes regras:

Pronomes/Advérbios - Quando um pronome ou advérbio é encontrado na estrutura sintática, um novo referente do discurso é adicionado e a condição de igualdade entre este referente e os referentes do discurso, que lhe estão relacionados e já definidos, também é adicionada. Todas as frases nominais que já tenham sido identificadas são possíveis candidatos para referentes do discurso que estão relacionados. Para os WH-pronomes e WH-advérbios que identificam o tipo de questão é adicionado um novo referente, com prefixo o tipo de pronome, e criado uma condição que identifica o tipo de questão em causa.

A configuração sintática é substituída pelo novo referente.

Exemplo 4.3.1 Considere a questão

```
Who is Barbara Jordan?
```
(4.2)

com a seguinte estrutura sintática

```
(ROOT
  (SBARQ
    (WHNP (WP who))
    (SQ (VBZ is)
      (NP (NNP Barbara) (NNP Jordan)))
    (. ?)))
```

O pronome `who` gera um novo referente, por exemplo `X`, com adição do prefixo `who`, e a condição `person(X)`. Na nova representação da questão, a representação semântica, o referente e a condição substituem a correspondente estrutura sintática. ▲

Nomes Próprios - Quando um nome próprio é encontrado, um novo referente do discurso (Y) é adicionado e é criado a condição "name(Y,nome próprio)" e que substituem a correspondente configuração sintática do nome próprio. Quando existem vários nomes próprios na questão e estão localizados no mesmo grupo sintático, são agrupados e formam apenas um nome, i.e., considera-se apenas um nome constituído por várias palavras.

Na questão 4.2, os nomes "Barbara" e "Jordan" estão no mesmo agrupamento da estrutura sintática, pelo que é gerado um novo referente, por exemplo `Y`, e a condição `name(Y, 'Barbara Jordan')`. O novo referente é adicionado à estrutura semântica da questão e a condição criada substitui a correspondente estrutura sintática.

Verbos – Quando um verbo é encontrado, é criada uma condição, formada pelo nome do verbo e com tantos argumentos como os referentes já definidos, que substitui a configuração sintática que ativa esta regra.

No exemplo para a questão (4.2), o verbo `is` é utilizado para definir a condição `is(X,Y)`, onde os argumentos são os referentes já criados e que estão relacionados no discurso.

Regra Geral - Todas as palavras identificadas na estrutura sintática, quando traduzidas para a correspondente estrutura semântica, são reescritas na forma normal e singular. No caso dos verbos, estes são traduzidos para o presente do indicativo, facilitando a procura ou representação de termos semelhantes.

Raiz - A marcação `ROOT` identifica a raiz da estrutura sintática de uma questão/frase, na estrutura semântica será substituída pala marcação `drs`.

Exemplo 4.3.2 Voltando à questão 4.1, apresentada no Exemplo 4.2.1, a Interpretação Semântica tem como função reescrever a estrutura sintática da questão, de acordo com o conjunto de regras exposto, e integrá-la na correspondente estrutura semântica, na forma de DRS.

De acordo com as regras apresentadas:

- O determinante "all" gera um novo referente `Y`, com prefixo `all`, e todos os pronomes e adjetivos que estão no mesmo agrupamento formam novas condições sobre o referente, nomeadamente `writer(Y)`, `french(Y)`, `romantic(Y)`. O termo "writers" da estrutura sintática é reescrito na sua forma normal e singular;
- O verbo "have" gera um novo referente `X`, com prefixo `have`, e é criada a condição de igualdade `is(X,Y)`, cujos argumentos são os referentes criados e incluídos no seu grupo verbal;
- O verbo "died" é traduzido para a sua forma normal e singular "die" e gera uma condição sobre o referente do seu agrupamento, `X`, a condição `die(X)`.

Todas os referentes gerados e todas as condições formadas substituem as configurações sintáticas, que ativaram as regras criadas. Assim, a DRS da questão apresentada é estabelecida pela seguinte estrutura de representação:

```
  U   =  {have-X, all-Y}
Con   =  {writer(Y), french(Y), romantic(Y), die(X), is(X,Y)}
```

onde os referentes do discurso são `have-X` e `all-Y`, com `X` uma entidade do discurso existencialmente quantificada e `Y` uma entidade do discurso universalmente quantificada; o predicado principal da questão é `is(X,Y)`; e os predicados pressupostos são `writer(Y)`, `french(Y)`, `romantic(Y)` e `die(X)`.

Os próximos passos do sistema será encontrar e validar, para as entidades `X` e `Y` que verificam todas as condições dos pressupostos da questão, se todas as entidades `Y` são também entidades `X`, condição verificada pelo predicado principal da questão. Se esta constatação for verdadeira, a resposta à questão inicial será afirmativa e, de forma a fornecer uma resposta mais informativa, o sistema também apresentará uma lista de todas as entidades que verificam a questão, i.e., as entidades francesas, escritoras e românticas que morreram. ▲

A Interpretação Semântica também tem como função classificar as questões de acordo com o seu tipo e com o tipo de resposta esperada, de forma a ajudar o sistema a fornecer uma resposta mais próxima das pretensões do utilizador.

4.3.1 Classificação das Questões

A classificação das questões é necessária para processar e julgar a questão.

De acordo com Manfred Krifka [63] é possível distinguir três tipos de questões, de acordo com o tipo de informação em falta: questões constituintes, questões de polaridade e questões alternativas. As questões constituintes criam uma proposição aberta, por deixar as partes da descrição da proposição não especificadas. Algumas línguas naturais aplicam a pro forma interrogativa para este propósito. Na língua Inglesa, estas interrogativas são inicializadas pelo termo "wh-(pronome ou questão)". Mais concretamente são as questões que começam com os termos "what", "who", "when", "where", "why", "which" e "whom". As questões de polaridade são também denominadas por questões "yes/no". Finalmente, as questões alternativas estão semanticamente relacionadas com as questões constituintes, diferindo destas no facto de mencionarem explicitamente os possíveis complementos.

Uma década antes, Hirschman e Gaizauskas [46] apresentaram uma forma possível de distinguir as questões através do tipo de respostas: respostas factuais, respostas de opinião e respostas de descrição (sumário). Consideraram ainda que é possível classificar as questões dependendo do seu tipo: questões "yes/no"; questões "wh" (*Who is Anubis*, *Where was John Adams born*); pedidos indiretos (*I would like you to list ...*); e listas (ordenações) (*Name all the presidents ...*).

Por análise das diferentes abordagens feitas, é possível identificar um conjunto de questões que indicam claramente o tipo de resposta esperada:

- "who/whom" - sugere que a resposta corresponda a uma pessoa, a um nome de companhia ou a um título profissional;

- "when" - sugere que a resposta seja uma data;
- "where" - sugere que a resposta seja uma localização, um lugar;
- "which" - refere-se a coisas inanimadas e a animais;
- "what" - refere-se a factos.

No presente trabalho, as questões são classificadas de acordo com as suportadas pelo sistema, incluindo: questões básicas com resposta esperada "yes/no" e as questões (do tipo "wh") "who", "where", "when", "what", e "which", respetivamente por "PERSON/ORGANIZATION", "PLACE/LOCATION", "DATE/TIME", "THING/OBJECT", "CHOICE".

Concluída a execução do módulo Interpretação Semântica, a informação sobre a classificação da questão e a representação semântica da questão faz parte do domínio do conhecimento sobre a questão, e será utilizada nos próximos passos do sistema.

Exemplo 4.3.3 A questão 4.1, considerada no exemplo 4.2.1, é uma questão do tipo "yes/no". No final, o resultado da interpretação semântica, aplicada à análise sintática da questão, é constituído pelo seguinte conjunto de informação que será adicionado ao domínio do conhecimento da questão em causa:

```
Question Type  =  yes/no
            U  =  {have-X, all-Y}
          Con  =  {writer(Y), french(Y), romantic(Y), die(X), is(X,Y)}
```

▲

4.4 Contexto Ontológico

O Contexto Ontológico é orientado pelo Controlador do Discurso de forma a obter as extensões da representação semântica da questão, no decorrer do processo de raciocínio, e é invocado quando o Controlador do Discurso tem de procurar pelas entidades da base de conhecimento que representam os conceitos da questão. O contexto de raciocínio e o significado da questão poderão ser atualizados sempre que o Controlador do Discurso chegue a um impasse.

Nesta fase, o sistema procura recursos na ontologia que sejam semelhantes (ou estejam relacionados) com os termos associados aos predicados da DRS. Para tal, faz a verificação de semelhanças entre os rótulos dos termos e recursos envolvidos, baseada nas suas composições como strings, tendo em conta abreviaturas, acrónimos, o domínio do conhecimento e o conhecimento léxico. Para maximizar a recolha, o Contexto Ontológico procura por

classes, propriedades, instâncias e/ou valores de dados que tenham rótulos que correspondam total ou parcialmente a um termo de procura da DRS.

Se uma entidade na procura não for encontrada, o termo da questão é estendido aos seus sinónimos, hiperónimos e hipónimos obtidos da WordNet [113], sempre com o objetivo de se encontrar informação na ontologia que possa representar os termos da questão. No final, obtém-se um conjunto de recursos semânticos, recursos da ontologia que correspondem aos termos da questão, e que podem conter a informação necessária para responder à questão. Se a representação na ontologia de um termo da questão e seus derivados não for encontrada, o Controlador do Discurso é alertado e é iniciado um diálogo controlado com o utilizador. O utilizador pode clarificar o sistema, reformulando a questão ou apresentar outras questões.

Exemplo 4.4.1 Voltando à questão 4.1, de forma a obter a extensão da representação da questão ao longo do processo de raciocínio, o módulo Contexto Ontológico tem de encontrar as classes, as propriedades e/ou os recursos, na base de conhecimento que tenham rótulos coincidentes ou contenham os termos da pesquisa "writer", "french", "romantic" e "die". Ou seja, o sistema tem de encontrar resposta às seguintes questões:

- Quais as Classes, Propriedades e/ou Instâncias que representam o conceito "writer"?

  O sistema encontra a propriedade da DBpedia `Writer`[4], com domínio `Work`[5] e âmbito `Person`[6]. Estes domínios informam o sistema sobre as propriedades das classes e podem confirmar quando estão relacionadas com os termos da questão, se não estão, serão deitados fora e uma nova pesquisa será feita. Por exemplo, na fase da interpretação gramatical, um dos pressupostas da questão estabelece que as entidades que são solução da questão terão de ser pessoas. Assim, se a classe `Writer`, não tem qualquer relação com a classe `Person`, ou não puder ser aplicado a pessoas, na fase da Interpretação Semântica não poderá ser adicionada ao conjunto de factos que representam a informação fornecida pela questão e, por conseguinte, não será considerada aquando da construção da resposta;

- Quais as Classes, Propriedades e/ou Instâncias que representam o conceito "french"?

  A DBpedia contém uma classe `birthPlace`[7] (uma entidade do tipo `ObjectProperty`[8], com domínio `Person` e âmbito `Place`[9]) que representa o lugar onde uma pessoa nasceu. O termo "french" também é interpretado como sendo "person of France" e tem como hiperónimo direto o termo "country" (obtido da WordNet), pelo que o sistema também tem de encontrar as classes, propriedades e/ou instâncias de todos

[4] `http://dbpedia.org/ontology/writer`
[5] `http://dbpedia.org/ontology/Work`
[6] `http://dbpedia.org/ontology/Person`
[7] `http://dbpedia.org/ontology/birthPlace`
[8] `http://www.w3.org/2002/07/owl#ObjectProperty`
[9] `http://dbpedia.org/ontology/Place`

os significados similares dos termos que compõem a questão e que poderão levar à resposta correta;

- Quais as Classes, Propriedades e/ou Instâncias que representam o conceito "romantic"?

  O sistema encontra o recurso da DBpedia `Romanticism`[10] (uma entidade do tipo `Thing`[11] e um valor da propriedade `movement`[12]);

- Quais as Classes, Propriedades e/ou Instâncias que representam o conceito "die"?

  A DBpedia tem a classe `deathDate`[13] (uma entidade do tipo `DatatypeProperty`[14], com domínio `Person` e no âmbito `date`[15]) que representa a data em que uma pessoa morreu. A relação entre os termos "die" e "death" pode ser estabelecida através da pesquisa na WordNet, onde o termo "die" pode ser interpretado como "decease", que por sua vez tem como sinónimo o termo "death".

O predicado principal `is` é interpretado, pelo Contexto Ontológico, como uma relação de igualdade entre duas entidades, no contexto da base de conhecimento. Ou seja, a relação não se limita a verificar e estabelecer a igualdade entre duas entidades, mas também contempla a relação de inclusão e a relação de similaridade. As entidades devolvidas pela pesquisa na ontologia poderão ser recursos da DBpedia, valores numéricos ou strings. Pelo que, o predicado `is(X,Y)` representa:

- `X == Y`, quando as duas entidades `X` e `Y` são a mesma;
- `X` $\subseteq$ `Y`, quando `X` é uma string e está relacionada com a entidade `Y`;
- `Y` $\subseteq$ `X`, quando `Y` é uma string e está relacionada com a entidade `X`.

▲

Concluída a procura na ontologia, o próximo passo consiste em interpretar e avaliar a informação recolhida, de acordo com a semântica da questão, e se não estiverem relacionadas com a questão, a procura por recursos na ontologia continuará.

[10] `http://dbpedia.org/resource/Romanticism`
[11] `http://www.w3.org/2002/07/owl#Thing`
[12] `http://dbpedia.org/property/movement`
[13] `http://dbpedia.org/ontology/deathDate`
[14] `http://www.w3.org/2002/07/owl#DatatypeProperty`
[15] `http://www.w3.org/2001/XMLSchema#date`

4.5 Avaliação Semântica

A Avaliação Semântica é o mecanismo que permite restringir a representação semântica inicial da questão, através da adição de novas condições que limitam as entidades do discurso associadas à questão. Esta fase caracteriza-se como sendo a avaliação pragmática do sistema, onde a representação semântica da questão é transformada num problema de satisfação de restrições.

Definição 4.5.1 *A **Avaliação Pragmática** é a capacidade de julgar ou calcular a qualidade, a importância, o valor das soluções do problema obtidas pela resolução real do problema, adaptado às condições atuais em vez de obedecer a teorias, ideias fixas ou regras.*

A transformação da representação semântica da questão num problema de restrições é conseguida através da adição de condições que limitam as entidades do discurso. O Problema de Satisfação de Restrições é constituído por um conjunto de regras de inferência, traduzidas em Prolog, que permite verificar a consistência dos termos da ontologia obtidos e encontrar a(s) solução(ões) para a questão colocada pelo utilizador. Ou seja, os predicados da DRS são interpretados como simples factos Prolog, onde as restrições do problema são os predicados da representação semântica e as variáveis do problema são as entidades associadas aos referentes da DRS. As soluções do problema de satisfação de restrições representam o conjunto de factos que suportam a resposta à questão colocada pelo utilizador.

A Avaliação Semântica tem de reinterpretar a representação semântica da questão, baseada na ontologia considerada, de forma a obter o conjunto de informação que verifica a questão. Ou seja, o processo responsável pela Avaliação Semântica reinterpreta a DRS da questão, mapeada na ontologia, de acordo com a base de conhecimento e com regras derivadas da ontologia. Este módulo do sistema é executado quando as representações da DRS na ontologia são encontradas, pelo módulo Contexto Ontológico. Pelo que, o sistema tem de encontrar os recursos/entidades da base de conhecimento que verificam cada representação na ontologia da DRS, i.e., que são solução da questão. As soluções encontradas, conjuntamente com as representações na ontologia da DRS que deram origem a soluções são adicionadas à base de conhecimento. O módulo da Avaliação semântica faz uso da ontologia, técnicas semânticas baseadas em lógica, inferência lógica e da consulta SPARQL.

Definição 4.5.2 *A correspondência entre um termo, expresso em língua natural, e uma entidade da ontologia denomina-se **mapeamento na ontologia**. Considere* `ontology_name` *a entidade da ontologia que corresponde ao mapeamento atribuído a um termo* `name`*, expresso em língua natural. Em particular, o mapeamento na ontologia do termo* `writer` *é a entidade* `ontology_writer`*.*

Se na base de conhecimento não existirem entidades que verifiquem o problema de satisfação de restrições, o Controlador do Discurso terá de continuar a sua pesquisa na obtenção de termos semelhantes, invocando novamente Contexto Ontológico. O processo de procura e avaliação continuará até se conseguir fornecer uma resposta ao utilizador. Sempre que o sistema detetar casos ambíguos (tais como: não conseguir decidir na escolha entre dois ou mais conceitos; não encontrar informação na base de conhecimento que represente os termos; ou obter respostas múltiplas), inicia um diálogo controlado com o utilizador de forma a obter informações que permitam esclarecer e continuar o processo na busca da resposta.

Exemplo 4.5.1 Voltando à questão 4.1 e que tem sido utilizada como ilustração da execução dos diferentes módulos do sistema, nesta fase o sistema começará por ter de encontrar as soluções que verificam os pressupostos da questão inicial. Ou seja, o sistema tem de encontrar as soluções para um novo conjunto de questões, tais como

```
Who are the French romantic writers?
```
(4.3)

com estrutura sintática, fornecida pelo analisador *Stanford*:

```
(ROOT
  (SBARQ
    (WHNP (WP Who))
    (SQ (VBP are)
        (NP (DT the) (NNP French) (JJ romantic) (NNS writers)))
    (. ?)))
```

e por aplicação das regras de conversão, a estrutura semântica de representação é

```
  U   =  {wh-X, exist-Y}
Con   =  {writer(Y), french(Y), romantic(Y), person(X), is(X,Y)}
```

Primeiro e de acordo com o domínio do conhecimento, o módulo da Avaliação Semântica irá transformar as condições da DRS em condições OWL, i.e., constrói os correspondentes predicados baseados na ontologia. Se a Avaliação Semântica encontrar várias possibilidades de condições da ontologia para cada termo então obterá várias DRS reescritas com os termos da ontologia.

Por conseguinte, a estrutura de representação da questão 4.3, mapeada na ontologia, é

```
  U   =  {wh-X, exist-Y}
Con   =  {ontology_writer(Y), ontology_french(Y),
          ontology_romantic(Y), ontology_person(X), is(X,Y)}
```

Ou seja, esta etapa do sistema consiste na construção de um conjunto de questões, necessário para restringir a pesquisa e interpretação da questão inicial. Este conjunto de novas questões é obtido por utilização de um conjunto de regras de inferência, traduzidas em predicados Prolog, que permitem a verificação da consistência dos termos e entidades encontrados. Alguns dos predicados Prolog têm como função procurar os triplos RDF na DBpedia que permitam relacionar as entidades encontradas e, por conseguinte, validá-las. Por exemplo, as entidades `http://dbpedia.org/ontology/Writer` e `http://dbpedia.org/resource/Romanticism` estão relacionadas e podem ser validadas para a continuação do processo porque a base do conhecimento contém o recurso `http://dbpedia.org/resource/Victor_Hugo` que verifica os pressupostos da questão inicial e que aparece nos seguintes triplos[16] RDF da DBpedia

```
dbpedia:Victor_Hugo dbpedia-owl:movement dbpedia:Romanticism .
dbpedia:Victor_Hugo rdf:type dbpedia-owl:Writer .
```

A pesquisa é feita exaustivamente e com o objetivo de encontrar todas as entidades que verificam a questão 4.3. Estas entidades verificam os predicados pressupostos da questão inicial 4.1, i.e., são entidades que representam pessoas francesas, românticas e escritoras. Depois, o sistema tem de verificar, por aplicação do predicado principal `is`, quais destas entidades verificam o predicado `die`. As entidades resultantes são solução da questão inicial e serão adicionadas ao domínio do conhecimento. Por outro lado, se existir alguma entidade que não verifica o predicado `die`, mas verifica os predicados pressupostos, será adicionada ao domínio do conhecimento classificada como uma entidade que não verifica a questão inicial.

Quando o sistema completa o processo de pesquisa das soluções da questão inicial 4.1, produzirá uma resposta de acordo com o tipo de questão. A questão apresentada é do tipo "yes/no" e, uma vez que todas as entidades pessoas encontradas, que são simultaneamente francesas, escritoras e românticas também já morreram, a resposta à questão será afirmativa. Para providenciar uma resposta mais informativa e cooperativa, o sistema mostrará uma lista das entidades encontradas e que verificam a questão.

```
http://dbpedia.org/resource/François-René_de_Chateaubriand
http://dbpedia.org/resource/Alphonse_de_Lamartine
http://dbpedia.org/resource/Alfred_de_Musset
http://dbpedia.org/resource/Victor_Hugo
http://dbpedia.org/resource/Stendhal
```

▲

[16] Os prefixos da ontologia da DBpedia traduzem-se por:
`dbpedia: <http://dbpedia.org/resource/>`;
`dbpedia-owl: <http://dbpedia.org/ontology/>`;
`rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>`.

Se toda a informação recolhida, ao longo deste processo, estiver relacionada com a questão inicial, será acrescentada à base de conhecimento e posteriormente utilizada no processamento da resposta. Se o sistema não encontrar soluções para a questão inicial, um conjunto de questões similares é construído. Questionando a WordNet, o sistema obtém termos similares aos que compõem a questão inicial. Este conjunto de questões similares enriquecerá o domínio do conhecimento que ajudará na interpretação da questão inicial ou na construção da sua resposta. Se o conjunto de soluções levar o sistema a diferentes respostas, estamos na presença de uma ambiguidade e o utilizador será invocado para a clarificar. Se o sistema não encontrar qualquer correspondência na base do conhecimento para um termo ou seus derivados, o utilizador será informado e poderá reformular a questão ou colocar uma nova questão.

4.6 Controlador do Discurso

O Controlador do Discurso é a componente principal do sistema e é invocado após a questão em língua natural ter sido transformada na sua representação semântica DRS. Basicamente, o Controlador do Discurso tenta dar sentido à questão colocada pelo utilizador, por análise da estrutura da ontologia e da informação disponibilizada na Web Semântica, como também através da utilização da correspondência de similaridade entre strings e recursos genéricos lexicais, com o objetivo de fornecer uma resposta ao utilizador.

No próximo capítulo 5, é feita uma abordagem mais completa sobre o Controlador do Discurso. De forma sucinta, depois da questão, colocada em língua natural pelo utilizador, ser transformada na sua representação semântica, o Controlador do Discurso é invocado e controla todo o processo até ao fim, i.e., até ser possível fornecer uma resposta ao utilizador. Mais especificamente, o módulo Contexto Ontológico é invocado com o objetivo de obter a extensão da representação semântica da questão na base de conhecimento. Se a representação na ontologia de um termo não é encontrado, significa que não é possível encontrar uma resposta para a questão. O Controlador do Discurso é alertado e o utilizador é invocado para que possa esclarecer os termos ou reescrever a questão colocada. Quando a extensão da representação semântica está concluída, o Controlador do Discurso adiciona ao seu conhecimento um conjunto de recursos semânticos. De seguida o módulo Avaliação Semântica é invocado. Nesta etapa, a representação semântica da questão é traduzida no correspondente problema de satisfação de restrições, por adição de restrições às entidades do discurso. O conjunto de soluções deste problema contém as entidades que verificam a questão inicial e que será adicionado ao domínio do conhecimento. Se a interpretação de toda a informação conduzir o Controlador do Discurso a uma resposta vazia ou a várias interpretações (caso de múltiplas respostas), o utilizador será invocado para clarificar e poderá ser necessário re-invocar os módulos Contexto Ontológico e Interpretação Semântica. O processo é finalizado quando o Controlador do Discurso é capaz de retornar uma resposta à questão colocada pelo utilizador.

O Controlador do Discurso lida com o conjunto das entidades do discurso e é capaz de

processar a resposta à questão. Tem de verificar os pressupostos da questão; escolher os recursos do conhecimento a serem usados; decidir quando a resposta a uma questão é alcançada; decidir quando continuar com o processo com novos recursos do conhecimento. A decisão de quando relaxar uma questão de forma a justificar a sua resposta e quando e como se deve clarificar uma questão também é da responsabilidade deste módulo.

Na interpretação da questão, quando o Controlador do Discurso não contém no seu conhecimento informação suficiente para decidir entre dois ou mais termos ou relações, inicia um diálogo controlado com o utilizador de forma a esclarecer as suas dúvidas. A informação fornecida pelo utilizador será essencial para que o Controlador do Discurso obtenha a resposta correta para a questão colocada pelo utilizador. Por exemplo:

- Na questão "Where is the Taj Mahal?" [24], o nome "Taj Mahal" pode ser interpretado como um nome de um Mausoléu, um Casino Hotel ou um Restaurante Indiano e apenas o utilizador poderá esclarecer sobre o significado pretendido;
- Suponha que pretende saber a resposta à questão "When did Bush took office?". O texto "took office" é um conceito temporal que está associado à tomada de posse de um cargo político e define o contexto da questão. Neste contexto, o nome "Bush" não é suficiente para responder à questão colocada pelo utilizador e origina um impasse. O sistema não sabe, nem tem informação suficiente para distinguir a quem o utilizador se está a referir, se ao presidente "George H. W. Bush", se ao seu filho "George W. Bush". A abertura de um diálogo controlado com o utilizador poderá esclarecer qual o presidente que se está a referir. Suponhamos que o utilizador pretende saber a data de tomada de posse do filho "George W. Bush". Novamente, o sistema fica numa encruzilhada, "George W. Bush" tem dois mandatos logo duas tomadas de posse. O sistema terá novamente de abrir um diálogo controlado com o utilizador e pedir esclarecimento sobre qual mandato se refere, se o primeiro ou o segundo;
- Se o utilizador pretende saber informação sobre "jaguar" e coloca a questão "What is a jaguar?" ao sistema. O facto de o tipo da questão ser "what", a questão pode referir-se a qualquer coisa, uma pessoa, um animal, uma organização, uma data, um evento, etc. Neste contexto, a palavra "jaguar" é ambígua, uma vez que a sua interpretação leva o sistema a obter quatro significados diferentes - um animal (predador felino), um sistema operativo (Jaguar, Mac 10.2), uma consola de jogos de vídeo (sistema Atari), ou um modelo de carros (Jaguar Land Rover). Novamente, o sistema não tendo informação suficiente, para saber qual o conceito a que se está a referir o utilizador, apenas com um diálogo controlado poderá esclarecer as pretensões do utilizador;
- A questão "What is the biggest district in Portugal?" é um exemplo de expressões dependentes do contexto no domínio geográfico. Neste contexto, o adjetivo "big" torna-se ambíguo: refere-se ao tamanho de um distrito tanto relativamente à população como com respeito à área. No entanto, estas duas interpretações levam a

duas respostas distintas[17]: relativamente ao número de habitantes, a resposta é Lisboa com 2.250.533; relativamente à área, a resposta é Beja com 10.225 km^2. Mais uma vez, a abertura de um diálogo controlado com o utilizador esclarecerá as dúvidas do sistema e assim responde às pretensões do utilizador.

A interatividade e cooperação do Controlador do Discurso torna-o mais eficaz e eficiente, uma vez que estará mais próximo de obter a resposta pretendida pelo utilizador.

Outro aspeto importante do Controlador do Discurso é fornecer uma resposta informativa ao utilizador. A resposta deve ser o mais próximo possível da língua natural. Por exemplo, o sistema Pergunta-Resposta tem de responder "yes" ou "no" quando o utilizador lhe coloca a questão "Is Barack Obama the President of the USA?". Neste caso, a resposta será "yes". No entanto, a resposta deve ser mais informativa e esclarecedora para o utilizador. Alguns aspetos estão definidos em contextos temporais, mesmo que implicitamente, e a resposta deverá ser o mais clara e informativa possível. Por exemplo, o termo "President", no contexto da questão, é definido como o título de chefe de Estado em algumas repúblicas e tem uma duração associada ao mandato, uma data de início (data da eleição, data da tomada de posse), e uma data de término do mandato. Assim, a resposta à questão "Is Barack Obama the President of the USA?" deverá ser "Yes, Barack Obama is the actual President of USA", que é mais cooperativa e informativa do que um simples "yes".

Para os casos onde a resposta a uma questão do tipo "Yes/No" é um "No", o Controlador do Discurso devolverá uma resposta completa, esclarecendo a negação. Consideremos a questão "Have all the capitals of Europe more than 200,000 inhabitants?" que tem uma resposta negativa, o sistema constrói a resposta adequada, baseando-se no conjunto de entidades do discurso que não verificam a questão, esclarecendo o utilizador sobre a sua interpretação na base do conhecimento e retornará "No, 9 capitals of Europe have less than or equal to 200,000 inhabitants".

Se a resposta a uma questão colocada pelo utilizador tiver múltiplas soluções, estas serão agrupadas de acordo com o seu significado semântico, fornecendo uma resposta mais clara e cooperativa ao utilizador. Por exemplo, para a questão "Where is the Taj Mahal?", o utilizador é invocado para clarificar a ambiguidade surgida na questão: "Taj Mahal" refere-se a um Mausoléu, a um restaurante ou a um Casino Hotel; e considerando que o utilizador não é capaz de clarificar o sistema ou simplesmente ele pretende ver todas as possíveis respostas. Assim, quando o sistema tiver todas as respostas para todas as interpretações possíveis para a questão colocada pelo utilizador, o Controlador do Discurso não irá fornecer as respostas de forma aleatória, mas lista as respostas agrupadas de acordo com os seus significados semânticos. Depois de agrupadas as respostas serão mostradas pela ordem com que estão armazenadas no conjunto de soluções. Desta forma, uma possível resposta para a questão "Where is the Taj Mahal?" poderá ser:

```
Mausoleum Taj Mahal is in Agra, India
```

[17] http://en.wikipedia.org/wiki/Ranked_list_of_Portuguese_districts

```
Casino hotel Taj Mahal is in Atlantic City, NJ, USA
Indian Restaurant Taj Mahal is in New Farm, Brisbane, Australia
Indian Restaurant Taj Mahal is in 7315 3rd Ave., Brooklyn, NY, USA
```

O nosso sistema tem como principal objetivo a interação com o utilizador, na forma de diálogo controlado, para obter respostas mais cooperativas, informativas e concretas. O diálogo controlado é utilizado não só para clarificar problemas de ambiguidade, mas também para ajudar o sistema a encontrar o caminho para a resposta correta. Tornar o sistema de diálogo mais cooperativo aumenta a sua capacidade de chegar mais perto da resposta esperada pelo utilizador. Em muitos casos, o utilizador é o único que poderá ajudar o sistema na dedução e interpretação da informação.

4.7 Conclusão

Neste capítulo apresentámos um sistema cooperativo de Pergunta-Resposta para a Web Semântica que recebe questões expressas em Inglês e é capaz de retornar respostas cooperativas, também expressas em Inglês, obtidas dos recursos na Web Semântica. A nossa proposta inclui pesquisa em profundidade, o uso de ontologias e outros recursos web, tais como a WordNet e a DBpedia. O sistema tem a capacidade de dialogar com o utilizador quando é encontrada alguma ambiguidade e que não permite que o sistema avance na pesquisa da solução. No próximo capítulo, apresentaremos com mais detalhe a componente principal do sistema cooperativo de Pergunta-Resposta para a Web Semântica proposto, o Controlador do Discurso.

Capítulo 5

Controlador do Discurso

Neste capítulo, apresentamos um Controlador do Discurso que, através da análise da questão e do tipo de resposta esperada, permite fornecer respostas precisas às questões colocadas pelo utilizador em língua natural (atualmente utilizamos apenas o Inglês). O Controlador do Discurso representa a semântica das questões e a estrutura do discurso, que contempla as intenções do utilizador e o contexto das questões, o que permite adicionar a capacidade de lidar com múltiplas respostas e de fornecer respostas justificadas. O capítulo está organizado da seguinte forma. Na Secção 5.1, começamos por apresentar o Controlador do Discurso proposto. Posteriormente, na Secção 5.2, apresentamos o processo de extração das respostas às questões colocadas pelo utilizador. Na Secção 5.3, expomos o procedimento efetuado no processamento das respostas por parte do Controlador do Discurso. E para finalizar, na Secção 5.4, apresentamos as conclusões.

5.1 Introdução

O Controlador do Discurso faz parte integrante, constituindo a componente principal, do sistema cooperativo de Pergunta-Resposta para Ontologias OWL2 [89, 87, 88], apresentado de forma genérica no capítulo anterior, e é invocado após a questão, em língua natural (atualmente utilizamos apenas o Inglês), ter sido transformada na sua representação semântica DRS. Basicamente, o Controlador do Discurso tenta dar sentido à questão colocada pelo utilizador, através: da análise da questão e do tipo de resposta esperada; da análise da estrutura da ontologia e da informação estruturada disponível na web (tal como a DBpedia [5]); e da utilização da correspondência da similaridade entre duas strings e de

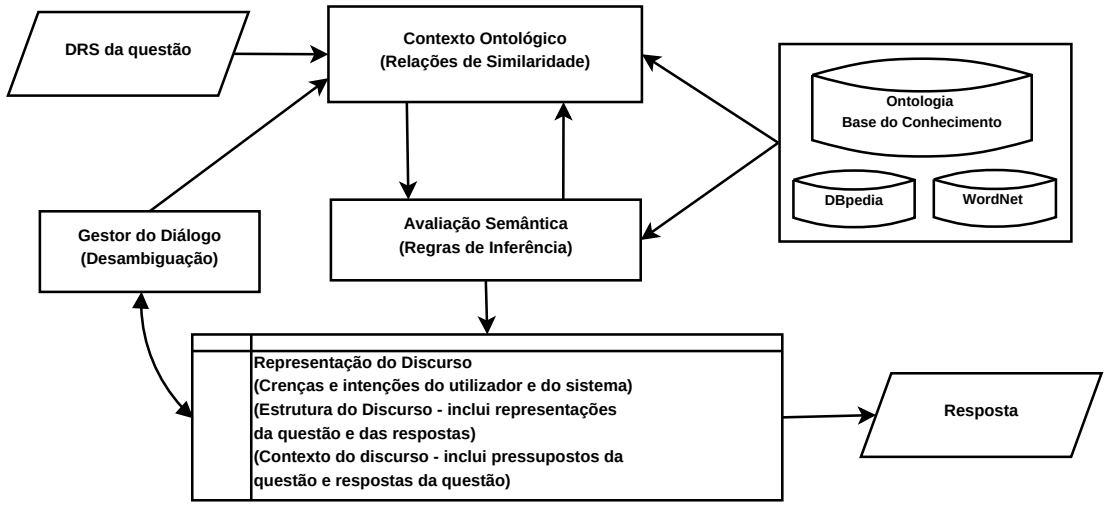


Figura 5.1: Arquitetura do Controlador do Discurso.

recursos lexicais genéricos (tais como a WordNet [33]), com o objetivo de fornecer uma resposta clara e informativa, ao utilizador.

Quando o Controlador do Discurso não é capaz de decidir o caminho correto para obter a resposta, inicia um diálogo controlado com o utilizador, com o objetivo de esclarecer as suas dúvidas. Esta ferramenta é independente do conhecimento prévio dos recursos semânticos, por parte do utilizador, e tem a capacidade de fornecer respostas diretas e precisas às questões colocadas pelo utilizador, em língua natural.

O Controlador do Discurso mantém a estrutura do diálogo, que fornece o contexto adequado para a interpretação das questões e para o processamento da resposta. Ou seja, o Controlador do Discurso lida com o conjunto das entidades do discurso e é capaz de processar a resposta à questão colocada pelo utilizador: verifica os pressupostos da questão para decidir quais os recursos do conhecimento (ontologias, WordNet, etc.) a serem utilizados; decide quando a resposta foi alcançada ou quando tem de avançar usando novos recursos do conhecimento. A decisão de quando aprofundar a resposta de forma a ser possível a sua justificação, quando ou como se deve clarificar uma questão, também são da responsabilidade deste módulo. Pelo que, o Controlador do Discurso representa as intenções e convicções tanto do sistema como do utilizador, a estrutura do discurso e o contexto da questão, incluindo contextos implícitos (tais como conhecimento temporal e espacial), entidades e informação útil para a interpretação semântica (tais como entidades do discurso usadas na resolução de anáforas ou como interpretar o significado de uma determinada instância de uma expressão, etc.), e que permitem acrescentar a capacidade de lidar com múltiplas respostas e de fornecer respostas justificadas.

A arquitetura do Controlador do Discurso é apresentada na Figura 5.1. De uma forma sucinta, o Controlador do Discurso é invocado após a questão, colocada pelo utilizador, ter sido transformada na sua representação semântica e controla todo o processo até ao fim, i.e., até o sistema ser capaz de retornar uma resposta ao utilizador. Mais especificamente, o

módulo Contexto Ontológico é invocado de forma a obter uma extensão da representação semântica da questão, na base de conhecimento. Quando a extensão da representação da questão está completa, o Controlador do Discurso adiciona ao seu conhecimento um conjunto de recursos semânticos e invoca o módulo Avaliação Semântica. Nesta etapa, a representação semântica da questão é traduzida no correspondente problema de satisfação de restrições, por adição de condições que restringem as entidades do discurso. O conjunto de soluções deste problema contém as entidades que verificam a questão inicial e que será adicionado ao domínio do conhecimento. Se a interpretação de toda a informação conduzir o Controlador do Discurso a uma resposta vazia ou a várias interpretações (caso de múltiplas respostas), o utilizador será invocado para clarificar e poderá ser necessário re-invocar os módulos Contexto Ontológico e Avaliação Semântica. O processo é finalizado quando o Controlador do Discurso é capaz de retornar uma resposta à questão colocada pelo utilizador.

5.2 Extração da Resposta

A extração da resposta consiste em encontrar todas as soluções para a questão colocada pelo utilizador. Ou seja, quando a questão em língua natural é transformada na sua representação semântica, o Controlador do Discurso tenta dar sentido à questão introduzida, através da análise da estrutura da ontologia e da informação disponível na Web Semântica, recorrendo a regras de similaridade entre strings e a recursos lexicais genéricos, com o objetivo de obter o conjunto de soluções que são resposta da questão.

O Controlador do Discurso deve supervisionar a pesquisa (por execução do módulo Contexto Ontológico) e validação (por execução do módulo Interpretação Semântica) das entidades na base do conhecimento e quando uma solução é encontrada será adicionada à base de conhecimento associada à questão inicial, mais concretamente, ao conjunto de informação que representa o discurso associado à questão colocada.

Exemplo 5.2.1 Considere a questão, apresentada em [24],

```
Where is the Taj Mahal?
```
(5.1)

que, de acordo com o analisador *Stanford*, tem como representação sintática a expressão

```
(ROOT
  (SBARQ
    (WHADVP (WRB Where))
    (SQ (VBZ is)
        (NP (DT the) (NNP Taj) (NNP Mahal)))
    (. ?)))
```

Por aplicação das regras introduzidas na Secção 4.3:

- Os nomes próprios "Taj" e "Mahal" estão no mesmo agrupamento da estrutura sintática, pelo que é gerado um novo referente, por exemplo `Y`, e a condição `name(Y, 'Taj Mahal')`. O determinante "the", por se tratar de um artigo definido e por preceder os nomes próprios será ignorado;
- O advérbio "where" com marcação de questão gera um novo referente, por exemplo `X`, e a condição `place(X)`. Para além disso, o advérbio "where" estando incorporado no agrupamento que faz ligação com os nomes próprios e implicitamente referir-se a uma localização, gera um novo referente, por exemplo `Z`, e a condição `location(Y,Z)`, cujos argumentos são os referentes criados e relacionados com o seu agrupamento;
- O verbo "is" gera a condição de igualdade `is(X,Z)`, cujos argumentos são os referentes criados e relacionados com o seu grupo verbal.

Assim, a representação semântica da questão 5.1 é expressa por:

$$\begin{aligned} \texttt{U} &= \{\texttt{where-X, exist-Y, exist-Z}\} \\ \texttt{Con} &= \{\texttt{name(Y,``Taj Mahal''), location(Y,Z), place(X), is(X,Z)}\} \end{aligned} \tag{5.2}$$

onde os referentes do discurso são `where-X`, `exist-Y` e `exist-Z`, com `X` uma entidade do discurso universalmente quantificada e `Y` e `Z` entidades do discurso existencialmente quantificadas, o predicado principal é `is(X, Z)` e os predicados pressupostos são `name(Y, 'Taj Mahal')`, `location(Y, Z)` e `place(X)`.

Na fase da execução do módulo Contexto Ontológico, o sistema tem de encontrar as entidades da base de conhecimento, que estão relacionadas com os termos da questão. Ou seja, o sistema procura recursos na ontologia que sejam semelhantes (ou estejam relacionados) com os termos associados aos predicados da DRS. Para tal, faz a verificação de semelhanças entre os rótulos dos termos e recursos envolvidos, baseada nas suas composições como strings, tendo em conta abreviaturas, acrónimos, o domínio do conhecimento e o conhecimento léxico. Para maximizar a recolha, o Contexto Ontológico procura por classes, propriedades, instâncias e/ou valores de dados que tenham rótulos que correspondam total ou parcialmente a um termo de procura da DRS. A Tabela 5.1 mostra alguns dos recursos encontrados e que estão relacionados com os termos da questão.

Os recursos da ontologia encontrados vão formar as representações DRS da questão na ontologia e para cada uma das representações terá de ser verificada a sua veracidade na base de conhecimento. Ou seja, por aplicação do módulo Avaliação Semântica, a validação de cada representação da DRS na ontologia consiste em encontrar todas as entidades da ontologia que são soluções da DRS. Mais concretamente, o sistema tem de encontrar os valores para as variáveis `X, Y, Z` que verificam as representações DRS na ontologia, i.e., para as correspondências encontradas no Contexto Ontológico, o sistema tem de encontrar as entidades/recursos da ontologia que verificam os triplos RDF que

Termos da DRS	Recursos da Ontologia
location	http://dbpedia.org/ontology/location
	http://dbpedia.org/ontology/locationCity
	http://dbpedia.org/ontology/locationCountry
	http://dbpedia.org/property/locationOfBirth
	http://dbpedia.org/property/locationArea
	http://dbpedia.org/property/locationmap
place	http://dbpedia.org/ontology/Place
	http://dbpedia.org/ontology/PopulatedPlace
	http://dbpedia.org/property/birthdplace
	http://dbpedia.org/ontology/deathPlace
Taj Mahal	http://dbpedia.org/resource/Taj_Mahal
	http://dbpedia.org/resource/Taj_Mahal_(musician)
	http://dbpedia.org/resource/Taj_Mahal_Palace_&_Tower
	http://dbpedia.org/resource/United_Breweries_Group
	http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)
	http://dbpedia.org/resource/Taj_Mahal_Palace
	http://dbpedia.org/resource/Taj_Mahal_(1963_film)
	http://dbpedia.org/resource/Taj_(album)
	http://dbpedia.org/resource/Mule_Bone_(album)
	http://dbpedia.org/resource/Taj_Mahal_Palace

Tabela 5.1: Alguns recursos da ontologia que correspondem a termos da DRS da questão "Where is the Taj Mahal?".

correspondem às condições da DRS 5.2 da questão. A solução e cada um dos triplos que verificam a estrutura semântica são adicionados à representação do discurso, a base de conhecimento da questão.

Por exemplo, para o recurso `http://dbpedia.org/resource/Taj_Mahal_Palace` que verifica a condição `name(Y,'Taj Mahal')`, encontramos na DBpedia as seguintes declarações[1] (triplos RDF), que traduzem, respetivamente, a sua localização (validação da condição `location(Y,Z)`) em Bhopal e que Bhopal se trata de um lugar (validação da condição `place(X)`):

```
dbpedia:Taj_Mahal_Palace dbpedia-owl:location dbpedia:Bhopal .
dbpedia:Bhopal rdf:type dbpedia-owl:Place .
```

[1]Os prefixos da ontologia da DBpedia traduzem-se por:
`dbpedia: <http://dbpedia.org/resource/>`;
`dbpedia-owl: <http://dbpedia.org/ontology/>`;
`rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>`.

Ou seja, estas declarações validam os mapeamentos dos termos da questão na ontologia, nomeadamente `location` é mapeado em `http://dbpedia.org/ontology/location` e `place` é mapeado em `http://dbpedia.org/ontology/Place`, e determinam uma solução para a representação semântica da questão

```
X = http://dbpedia.org/resource/Bhopal
Y = http://dbpedia.org/resource/Taj_Mahal_Palace
Z = http://dbpedia.org/resource/Bhopal
```

A solução, os triplos RDF que geram a solução e os mapeamentos dos termos da questão na ontologia, que validam a representação semântica da questão, são adicionados à base de conhecimento da questão, a representação do discurso. ▲

5.3 Processamento da Resposta

O Processamento da Resposta consiste na determinação da representação final da resposta que será retornada ao utilizador, que é interpretada na base de conhecimento com os factos extraídos. Nesta fase, o Controlador do Discurso analisa os factos obtidos (que são soluções da questão) e fornece ao utilizador uma resposta adequada, tendo em conta o tipo da questão e a resposta esperada.

- Se o conjunto de soluções é vazio, a resposta terá de informar o utilizador desse facto, que por sua vez poderá reescrever a questão inicial ou efetuar uma nova questão ou, simplesmente, terminar o processo.
- Se o conjunto de soluções tiver apenas uma solução, a resposta apresentada ao utilizador, para além de direta e objetiva, também informa sobre as entidades que serviram de suporte, permitindo uma melhor comunicação entre o sistema e o utilizador. Assim, o Controlador do Discurso apresenta as entidades da solução e respetivos sumários descritivos fornecidos pela DBpedia.
- Se o conjunto de soluções contempla múltiplas soluções, várias interpretações podem ser feitas. Se o Controlador do Discurso não tem informação suficiente para decidir qual é a correta, inicia um diálogo controlado com o utilizador. Pelo que, apresenta ao utilizador um conjunto de alternativas e a resposta dada pelo utilizador, a essas alternativas, permite clarificar a qual assunto se está a referir. O conjunto de alternativas é constituído por características que distinguem as soluções. O Algoritmo 1 mostra como o sistema processa a clarificação da ambiguidade respeitante à multiplicidade de soluções. Para o conjunto de soluções é construído um conjunto de alternativas A baseado nos valores da melhor propriedade e que caracteriza de forma distinta as diversas soluções. O conjunto de alternativas é apresentado ao utilizador. Quando a escolha do utilizador clarificar a ambiguidade o sistema pode fornecer a resposta esperada à questão inicial.

Algoritmo 1 Clarificação das intenções do utilizador

Entrada: $S = \{s|s$ é uma solução da questão$\}$
Saída: Definir uma resposta à questão
1: **Enquanto** $\#S > 1$ **faz**
2: Para cada referente colecionar as suas propriedades
3: Avaliar qual a melhor propriedade para diferenciar os referentes
4: Escolher a melhor propriedade
5: A = valores da melhor propriedade para cada solução (onde está definida a propriedade)
6: Mostrar as alternativas de clarificação baseadas no conjunto A
7: Receber escolha do utilizador
8: Restringir o conjunto de soluções S à escolha do utilizador
9: **Fim Enquanto**
10: Mostrar a solução S ao utilizador.

A Avaliação Semântica pode induzir o sistema a determinar múltiplos resultados, refletindo ambiguidades na questão. Nos casos em que o sistema não contém na sua base de conhecimento informação suficiente para esclarecer as ambiguidades que surgem, o sistema necessita de clarificar as intenções do utilizador utilizando, para tal, um diálogo controlado. Desta forma, o Controlador do Discurso ativa o mecanismo de clarificação através da execução do Algoritmo 1 que permite clarificar as intenções do utilizador, relativamente à sua questão inicial e às soluções obtidas pelo sistema. A reformulação deste algoritmo segue a ideia apresentada pelos autores de [101].

De forma a explicar a aplicação do Algoritmo 1, nas próximas secções faremos uma exposição detalhada dos seus principais passos e das metodologias utilizadas.

5.3.1 Avaliação das Propriedades

As alternativas de clarificação a apresentar ao utilizador devem cumprir dois importantes aspetos: informar o mais possível sobre as intenções do utilizador; e terem informação que com maior probabilidade seja conhecida pelo utilizador. O segundo aspeto tem por base a parametrização de que o utilizador poderá conhecer mais facilmente informação textual do que numérica. Por exemplo, se a clarificação diz respeito a características de uma pessoa, muito provavelmente, é mais fácil o utilizador conhecer o seu país do que a sua data de nascimento. Outra parametrização possível seria considerar que o utilizador conheceria melhor informação numérica.

Relativamente ao primeiro aspeto, a escolha das alternativas a apresentar ao utilizador consiste em escolher a melhor propriedade que contém mais quantidade de informação. A escolha da melhor propriedade baseia-se no modelo de Árvores de Decisão de Aprendizagem [100], mais concretamente o algoritmo ID3 (*Inductive Decision Tree*) utilizado como método de classificação na construção das árvores de decisão. Esta escolha teve como principal motivo o facto de as árvores de decisão poderem ser aplicadas a grandes

Nome	Recurso da Ontologia
Taj Mahal	`http://dbpedia.org/resource/Taj_Mahal`
Taj Mahal Palace & Tower	`http://dbpedia.org/resource/Taj_Mahal_Palace_&_Tower`
Taj Mahal Palace	`http://dbpedia.org/resource/Taj_Mahal_Palace`
Taj Mahal Hotel (Delhi)	`http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)`

Tabela 5.2: Recursos da base de conhecimento, entidades das soluções da questão "Where is the Taj Mahal?" que geram ambiguidade.

conjuntos de dados e possibilitarem uma visão real da natureza do processo de decisão. Para além disso, as árvores de decisão são um dos modelos mais práticos e mais usados em inferência indutiva. Este método representa funções como árvores de decisão. Estas árvores são treinadas de acordo com um conjunto de treino (exemplos previamente classificados) e, posteriormente, outros exemplos são classificados de acordo com essa mesma árvore.

Genericamente, uma árvore de decisão é representada da seguinte forma:

- Cada nó de decisão contém um teste numa propriedade;
- Cada ramo descendente corresponde a um possível valor desta propriedade que, em conjunto, correspondem às alternativas a serem apresentadas ao utilizador.

O algoritmo ID3 foi o primeiro algoritmo a ser utilizado em indução de árvores de decisão. É um algoritmo recursivo, que procura o conjunto de propriedades que melhor dividem os exemplos, no momento da escolha, gerando sub-árvores. O algoritmo ID3 utiliza entropia e ganho de informação para construir a árvore de decisão. No entanto, a classificação das propriedades por maximização do ganho de informação dá preferência a propriedades com muitos valores. Pelo que, introduzimos a razão do ganho também como critério de avaliação, favorecendo as propriedades cujo valor da entropia é pequeno e, por conseguinte, favorecendo as propriedades que contêm menor número de valores. Assim, para ajudar na tarefa da classificação e da escolha da melhor propriedade, cujos valores serão apresentados ao utilizador como alternativas de clarificação, introduzimos de seguida os três novos conceitos: a Entropia, o Ganho e a Razão do Ganho.

5.3.1.1 Entropia

A entropia de um conjunto pode ser definida como sendo o grau de pureza (certeza, precisão) desse conjunto. Este conceito emprestado pela Teoria da Informação define a medida de "falta de informação", mais precisamente o número de bits necessários, em média, para representar a informação em falta, usando codificação ótima. Se o conjunto

é completamente homogéneo, o valor da entropia é zero; e se o conjunto é dividido de forma equitativa, o valor da entropia é igual a 1. Formalmente, a entropia de um conjunto define-se do seguinte modo:

Definição 5.3.1 *Dado um conjunto T, com instâncias pertencentes à classe i, com probabilidade $p_i \neq 0$, a entropia do conjunto T é obtida pela seguinte expressão*

$$Entropia(T) = -\sum p_i \times log_2(p_i). \tag{5.3}$$

A entropia de um conjunto T verifica a propriedade $0 < Entropia(T) < log_2(n)$, onde n é o número total de classes i.

Exemplo 5.3.1 Voltando ao Exemplo 5.2.1, a Questão 5.1 colocada pelo utilizador refere-se à localização de "Taj Mahal". Quando o Controlador do Discurso analisa o conjunto de soluções deteta que está na presença de múltiplas respostas, uma vez que existem diferentes entidades que representam o termo "Taj Mahal", apresentados na Tabela 5.2.

O Controlador do Discurso não tendo informação suficiente para decidir qual das soluções é a correta, tem de iniciar um diálogo controlado com o utilizador, de forma que este possa conduzi-lo no caminho correto para fornecer a resposta que pretende. Para clarificar a situação de ambiguidade de ter múltiplas soluções, o Controlador do Discurso executa o Algoritmo 1.

Na DBpedia as propriedades são expressas por triplos RDF, pelo que o conjunto T é composto por todos os triplos, que formam as propriedades que estão associadas a cada um dos referentes da questão. No exemplo, excluímos o referente `X` que está associado ao advérbio da questão, pois trata-se de um referente que está semanticamente relacionado com o que o utilizador pretende saber. Por analogia, o referente `Z` também é excluído porque a condição `is(X,Z)` torna-o semanticamente igual ao referente `X`. Desta forma, resta-nos o referente `Y` que está associado ao nome "Taj Mahal". Por conseguinte, o conjunto T é constituído apenas pelas propriedades associadas ao referente `Y`.

A Tabela 5.2 apresenta os diferentes valores das soluções encontradas pelo sistema para o referente `Y`. Para estes valores, que constituem o conjunto de classes a que as instâncias de T pertencem, temos de formar o conjunto de todas as propriedades que lhes estão associadas (triplos RDF, na forma `Y Propriedade Valor`). Ao todo são 225 propriedades que formam o conjunto T. Na Tabela 5.3 é apresentado o cálculo de todas as parcelas, cada uma dependendo de uma classe i associada, que compõem a fórmula para determinar o valor da entropia do conjunto T. Assim, de acordo com a Definição 5.3.1 e os valores apresentados na Tabela 5.3, o valor da entropia do conjunto T é calculada da seguinte forma:

$$\begin{aligned} Entropia(T) &= \sum(-p_i \times log_2(p_i)) = \\ &= 0,5306152278 + 0,5041618283 + 0,4282672424 + 0,4789088711 = \\ &= 1,9419531697 \end{aligned}$$

Classe i	**#(Classe** i**)**	$\mathbf{p_i} = \frac{\#\mathbf{Classe}}{\#\mathbf{T}}$	$-\mathbf{p_i} \times \mathbf{log_2}(\mathbf{p_i})$
Taj Mahal	81	$\frac{81}{225} = 0,36$	$-\frac{81}{225} \times log_2(\frac{81}{547}) = 0,5306152278$
Taj Mahal Palace & Tower	58	$\frac{58}{225} = 0,2577777778$	$-\frac{58}{225} \times log_2(\frac{58}{225}) = 0,5041618283$
Taj Mahal Palace	37	$\frac{37}{225} = 0,1644444444$	$-\frac{37}{225} \times log_2(\frac{37}{225}) = 0,4282672424$
Taj Mahal Hotel (Delhi)	49	$\frac{49}{225} = 0,2177777778$	$-\frac{49}{225} \times log_2(\frac{49}{225}) = 0,4789088711$

Tabela 5.3: Cálculo de cada uma das parcelas que compõem a fórmula para determinar o valor da entropia do conjunto T.

e tem-se que $0 < Entropia(T) = 1,9419531697 < log_2(n) = log_2(4) = 2.$ ▲

5.3.1.2 Ganho de Informação

A construção de uma árvore de derivação é orientada pelo objetivo de diminuir a entropia, a dificuldade de previsão da variável que define as classes. O ganho de informação define a redução na entropia. De um modo mais formal temos:

Definição 5.3.2 *O ganho de informação é a redução esperada na entropia causada pela partição dos dados, de acordo com o teste na propriedade P. O valor do ganho de informação para a propriedade P é dado pela seguinte expressão:*

$$Ganho(T, P) = Entropia(T) - Entropia(T, P) \tag{5.4}$$

Para determinar o ganho de informação de uma propriedade temos de, para todos os valores da propriedade, calcular a entropia de cada valor e depois adicioná-las proporcionalmente de forma a obter a entropia total da propriedade. De um modo mais formal temos:

Definição 5.3.3 *Dado um conjunto T, com instâncias pertencentes à classe i, com probabilidade $p_i \neq 0$, a entropia total da propriedade P é obtida pela seguinte expressão*

$$Entropia(T, P) = \sum_{v \in valores(P)} (\frac{\#T_v}{\#T} \times Entropia(T_v)) \tag{5.5}$$

Exemplo 5.3.2 Continuando com o exemplo da Questão 5.1, no passo anterior calculamos o valor da entropia do conjunto T, que contém todas as propriedades associadas ao referente `Y` - referente que está na base da multiplicidade da resposta à questão. O ganho

de informação tem de ser calculado para todas as propriedades. Mais concretamente, para cada propriedade é calculada a entropia de cada valor distinto e depois são adicionadas proporcionalmente de forma a obtermos a entropia total da propriedade.

Como mencionámos anteriormente, o conjunto T tem 225 elementos, os triplos RDF das propriedades das entidades da solução associadas ao referente, que pode clarificar a multiplicidade de respostas à questão. Os elementos do conjunto T são formados por 74 propriedades distintas. Parecendo-nos uma tarefa ingrata e desnecessária expormos o cálculo do ganho de informação para todas as propriedades, apenas exemplificaremos para a propriedade que tem menor ganho de informação, a propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` e para uma das propriedades que tem maior ganho de informação, a propriedade `http://dbpedia.org/property/architecture`:

- A propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` ocorre 35 vezes no conjunto T, com diferentes valores para cada uma das classes - recursos mostrados na Tabela 5.2. Para determinar o ganho de informação desta propriedade temos de: primeiro, para todos os valores da propriedade calcular a entropia de cada valor e depois adicioná-las proporcionalmente de forma a obter a entropia total da propriedade; segundo, subtrair a entropia da propriedade à entropia dos dados, conjunto T. Assim, considere-se o conjunto T_1 que contém as ocorrências de T que correspondem à propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`. Considere-se ainda o conjunto $T(Valor)$ como sendo o conjunto de elementos de T_1 cuja propriedade tem valor igual a $Valor$. Existem 17 valores distintos, por conseguinte, apenas iremos expor para 3 casos e que permitem elucidar como o cálculo é feito.

  Considere $Valor$ = `http://dbpedia.org/class/yago/YagoGeoEntity`. O conjunto T_1 tem dois elementos onde ocorre esse valor, pelo que $\#T(Valor) = 2$. Nomeadamente os seguintes triplos

  ```
  <http://dbpedia.org/resource/Taj_Mahal>
        <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
              <http://dbpedia.org/class/yago/YagoGeoEntity> .


  <http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)>
        <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
              <http://dbpedia.org/class/yago/YagoGeoEntity> .
  ```

  De seguida, para cada classe destes triplos vamos determinar as parcelas necessárias para o cálculo da entropia do valor:

  i) Classe = `http://dbpedia.org/resource/Taj_Mahal`

$$p_i = \frac{\#Classe}{\#T(Valor)} = \frac{1}{2} = 0,5$$

$$-p_i \times log_2(p_i) = -\frac{1}{2} \times log_2(\frac{1}{2}) = 0,5$$

ii) Classe = `http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)`

$$p_i = \frac{\#Classe}{\#T(Valor)} = \frac{1}{2} = 0,5$$

$$-p_i \times log_2(p_i) = -\frac{1}{2} \times log_2(\frac{1}{2}) = 0,5$$

A entropia do valor `http://dbpedia.org/class/yago/YagoGeoEntity` é calculada somando proporcionalmente todos os valores anteriores:

$$\frac{\#T(Valor)}{\#T_1} \times (-\sum p_i \times log_2(p_i)) = \frac{2}{35} \times 0,5 + \frac{2}{35} \times 0,5 = 0,0571428571$$

Considere agora $Valor$ = `http://dbpedia.org/class/yago/MausoleumsInIndia`. O conjunto T_1 tem apenas um elemento onde ocorre esse valor, pelo que $\#T(Valor) = 1$. Nomeadamente o seguinte triplo

```
<http://dbpedia.org/resource/Taj_Mahal>
      <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
            <http://dbpedia.org/class/yago/MausoleumsInIndia> .
```

Pelo que

$$p_i = \frac{\#Classe}{\#T(Valor)} = \frac{1}{1} = 1$$

$$-p_i \times log_2(p_i) = -\frac{1}{1} \times log_2(\frac{1}{1}) = 0$$

Donde a entropia do valor `http://dbpedia.org/class/yago/YagoGeoEntity` é:

$$\frac{\#T(Valor)}{\#T_1} \times (-\sum p_i \times log_2(p_i)) = \frac{1}{35} \times 0 = 0$$

Finalmente, considere $Valor$ = `http://www.opengis.net/gml/_Feature`. O conjunto T_1 tem quatro elementos onde ocorre esse valor, pelo que $\#T(Valor) = 4$. Nomeadamente os seguintes triplos

```
<http://dbpedia.org/resource/Taj_Mahal>
```

```
        <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
              <http://www.opengis.net/gml/_Feature> .

<http://dbpedia.org/resource/Taj_Mahal_Palace_&_Tower>
        <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
              <http://www.opengis.net/gml/_Feature> .

<http://dbpedia.org/resource/Taj_Mahal_Palace>
        <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
              <http://www.opengis.net/gml/_Feature> .

<http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)>
        <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
              <http://www.opengis.net/gml/_Feature> .
```

De seguida, para cada classe destes triplos vamos determinar as parcelas necessárias para o cálculo da entropia do valor:

i) Classe = `http://dbpedia.org/resource/Taj_Mahal`

$$p_i = \frac{\#Classe}{\#T(Valor)} = \frac{1}{4} = 0,25$$

$$-p_i \times log_2(p_i) = -\frac{1}{4} \times log_2(\frac{1}{4}) = 0,5$$

ii) Classe = `http://dbpedia.org/resource/Taj_Mahal_Palace_&_Tower`

$$p_i = \frac{\#Classe}{\#T(Valor)} = \frac{1}{4} = 0,25$$

$$-p_i \times log_2(p_i) = -\frac{1}{4} \times log_2(\frac{1}{4}) = 0,5$$

iii) Classe = `http://dbpedia.org/resource/Taj_Mahal_Palace`

$$p_i = \frac{\#Classe}{\#T(Valor)} = \frac{1}{4} = 0,25$$

$$-p_i \times log_2(p_i) = -\frac{1}{4} \times log_2(\frac{1}{4}) = 0,5$$

iv) Classe = `http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)`

$$p_i = \frac{\#Classe}{\#T(Valor)} = \frac{1}{4} = 0,25$$

$$-p_i \times log_2(p_i) = -\frac{1}{4} \times log_2(\frac{1}{4}) = 0,5$$

A entropia do valor `http://www.opengis.net/gml/_Feature` é calculada somando proporcionalmente todos os valores anteriores:

$$\frac{\#T(Valor)}{\#T_1} \times (-\sum p_i \times log_2(p_i)) = \frac{2}{35} \times (0,5 + 0,5 + 0,5 + 0,5) = 0,2285714286$$

Fazendo o mesmo tipo de cálculo para os restantes valores da propriedade, a entropia da propriedade resulta da adição de todas as entropias dos seus valores. Assim,

$$\begin{aligned} &Entropia(T, \texttt{http://www.w3.org/1999/02/22-rdf-syntax-ns\#type}) = \\ &\quad = 0,0571428571 + 0 + 0,2285714286 + ... = \\ &\quad = 1,1935553575 \end{aligned}$$

Por conseguinte, e de acordo com a Expressão (5.4), o ganho de informação da propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` é

$$\begin{aligned} &Ganho(T, \texttt{http://www.w3.org/1999/02/22-rdf-syntax-ns\#type}) = \\ &\quad = Entropia(T) \\ &\quad\quad -Entropia(T, \texttt{http://www.w3.org/1999/02/22-rdf-syntax-ns\#type}) = \\ &\quad = 1,9419531697 - 1,1935553575 = \\ &\quad = 0.7483978122 \end{aligned}$$

- A propriedade `http://dbpedia.org/property/architecture` apenas ocorre uma vez no conjunto T, através do triplo

```
<http://dbpedia.org/resource/Taj_Mahal>
      <http://dbpedia.org/property/architecture>
            <http://dbpedia.org/resource/Mughal_architecture> .
```

pelo que o conjunto T_1 só contém este triplo. A entropia da propriedade `http://dbpedia.org/property/architecture` é dada por

$$\begin{aligned} &Entropia(T, \texttt{http://dbpedia.org/property/architecture}) = \\ &\quad = \frac{\#T(Valor)}{\#T_1} \times (-\sum p_i \times log_2(p_i)) = \\ &\quad = 0 \end{aligned}$$

Por conseguinte, e de acordo com a Expressão (5.4), o ganho de informação desta propriedade é

Propriedade	Entropia	Ganho de Informação	Razão do Ganho
http://purl.org/dc/terms/subject	0,0869565	1,855	21,3325053331
http://dbpedia.org/property/location	0,333333	1,60862	4,8258648259
http://dbpedia.org/ontology/location	0,333333	1,60862	4.8258648259
http://dbpedia.org/property/wikiPageUsesTemplate	0,5	1,44195	2,8839
http://dbpedia.org/property/latns	1	0,941953	0,941953
http://dbpedia.org/property/longd	1	0,941953	0,941953
http://dbpedia.org/property/longew	1	0,941953	0,941953
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	1,19356	0,748398	0.6270300613
http://dbpedia.org/property/architecture	0	1,94195	ND
http://dbpedia.org/property/built	0	1,94195	ND
http://dbpedia.org/property/coordDisplay	0	1,94195	ND
http://dbpedia.org/property/designation	0	1,94195	ND
http://dbpedia.org/property/designation1Number	0	1,94195	ND
http://dbpedia.org/property/designation1Type	0	1,94195	ND
http://dbpedia.org/property/elevation	0	1,94195	ND
http://dbpedia.org/property/imageSize	0	1,94195	ND
http://dbpedia.org/property/latitude	0	1,94195	ND
http://dbpedia.org/property/locmapin	0	1,94195	ND
http://dbpedia.org/property/longitude	0	1,94195	ND
...	...	...	...
http://dbpedia.org/property/latd	0	1,94195	ND
http://dbpedia.org/property/latm	0	1,94195	ND
http://dbpedia.org/property/lats	0	1,94195	ND
http://dbpedia.org/property/longm	0	1,94195	ND
http://www.georss.org/georss/point	0	1,94195	ND
http://xmlns.com/foaf/0.1/name	0	1,94195	ND
http://dbpedia.org/ontology/floorCount	0	1,94195	ND

Tabela 5.4: Resultados obtidos relativamente à Entropia, Ganho de Informação e Razão do Ganho de Informação das propriedades utilizadas para a clarificação da multiplicidade de soluções da questão "Where is the Taj Mahal?" - primeira iteração da aplicação do Algoritmo 1.

$$
\begin{aligned}
&Ganho(T, \texttt{http://dbpedia.org/property/architecture}) = \\
&\quad = Entropia(T) \\
&\qquad -Entropia(T, \texttt{http://dbpedia.org/property/architecture}) = \\
&\quad = 1,9419531697 - 0 = \\
&\quad = 1,9419531697
\end{aligned}
$$

A Tabela 5.4 apresenta os resultados obtidos relativamente à Entropia e Ganho de Informação de algumas propriedades, utilizadas para a clarificação da multiplicidade de soluções da questão "Where is the Taj Mahal?". Suprimimos algumas das propriedades que têm entropia igual a zero - das 74 propriedades existem 66 que têm o valor da entropia igual a zero. ▲

5.3.1.3 Razão do Ganho

O conceito Razão de Ganho de Informação de uma propriedade é nada mais do que o ganho de informação da propriedade relativo (ponderado) à entropia da propriedade. Ou seja,

Definição 5.3.4 *Considere-se a propriedade P de um conjunto T, o valor da razão do ganho de informação para a propriedade P no conjunto T é dado pela seguinte expressão:*

$$RazaoGanho(T, P) = \frac{Ganho(T, P)}{Entropia(T, P)} \tag{5.6}$$

A razão do ganho não é definida quando a $Entropia(T, P) = 0$.

Exemplo 5.3.3 Continuando com o exemplo da questão 5.1, no secção anterior calculámos o ganho de informação para cada uma das propriedade do conjunto T, propriedades associadas ao referente `Y` - referente que está na base da multiplicidade da resposta à questão. Para as propriedades selecionadas para ilustrar o cálculo do ganho de informação, iremos proceder ao respetivo cálculo da razão do ganho de informação. Assim, e de acordo com a expressão 5.6,

- Para a propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`, tendo em conta que

  $$Entropia(T, \texttt{http://www.w3.org/1999/02/22-rdf-syntax-ns\#type}) = 1,19356$$
  $$Ganho(T, \texttt{http://www.w3.org/1999/02/22-rdf-syntax-ns\#type}) = 0,748398$$

  a razão do ganho de informação é

  $$RazaoGanho(T, \texttt{http://www.w3.org/1999/02/22-rdf-syntax-ns\#type}) = \\ = 0.6270300613$$

- Para a propriedade `http://dbpedia.org/property/architecture`, tendo em conta que

  $$Entropia(T, \texttt{http://dbpedia.org/property/architecture}) = 0$$
  $$Ganho(T, \texttt{http://dbpedia.org/property/architecture}) = 1$$

  a razão do ganho de informação não é definida (ND).

A Tabela 5.4 apresenta também os resultados obtidos à Razão do Ganho de Informação, para além da Entropia e Ganho de Informação, de algumas propriedades , utilizadas para a clarificação da multiplicidade de soluções da questão "Where is the Taj Mahal?". ▲

5.3.2 Escolha da Melhor Propriedade

O critério de ganho de informação de uma propriedade seleciona como propriedade de teste aquela que maximiza o ganho de informação. No entanto, este critério dá preferência a atributos com muitos valores possíveis (que corresponde ao número de arestas da árvore de decisão). Um exemplo claro desse problema ocorreria ao utilizar um atributo totalmente irrelevante (por exemplo, um identificador único, à semelhança do que acontece com o exemplo da questão 5.1 que estamos a expor). Nesse caso, teríamos uma alternativa para cada valor possível, e o número de alternativas seria igual ao número de identificadores. Cada uma dessas alternativas teria apenas um exemplo, o qual pertence a uma única classe. Pelo que, o valor da entropia seria mínima porque, em cada propriedade, todos os exemplos (no caso um só) pertencem à mesma classe, que geraria um ganho máximo, embora totalmente inútil. Quando este problema acontece, i.e., quando para uma propriedade P num conjunto T se tem $Entropia(T, P) = 0$ e que corresponde ao ganho de informação máximo, utilizamos como solução a razão de ganho de informação para critério de avaliação.

No entanto, mesmo com estes critérios poderemos ter como melhor propriedade a escolher, uma propriedade cuja informação seja menos conhecida para o utilizador. Por exemplo, se a melhor propriedade representar valores numéricos (como datas de nascimento, nº de cartão do cidadão, etc.), supomos que o utilizador poderá não saber tais informações. Pelo que a apresentação de tal alternativas ao utilizador irá resultar num passo desnecessário. Assim, para estes casos definimos como prioritária as propriedades que contenham informação (valor) não numérica, i.e., que represente texto.

Definição 5.3.5 *Considere-se P uma propriedade de um conjunto T. P é a* ***melhor propriedade*** *a escolher, para apresentar os seus valores como alternativas no diálogo controlado com o utilizador, quando o valor da razão do ganho RazaoGanho(T,P) é o maior, de entre todos os valores da razão do ganho das propriedades de T, e P é uma propriedade cujo valor é expresso em texto.*

Exemplo 5.3.4 Voltando ao exemplo da questão 5.1 e de acordo com a Tabela 5.4, a propriedade com maior razão de ganho de informação é a propriedade `http://purl.org/dc/terms/subject`. Como esta propriedade tem valores não numéricos (expressos em texto), será a escolhida para apresentar os seus valores como alternativas no diálogo controlado com o utilizador. Caso os valores desta propriedade fossem numéricos, escolheríamos a próxima propriedade com melhor valor da razão do ganho de informação e cujos valores fossem expressos em texto. ▲

5.3.3 Diálogo Controlado com o Utilizador

Definida a forma como se faz a avaliação das propriedades - através da utilização da entropia, do ganho de informação e razão do ganho de informação; definida a forma como se escolhe a melhor propriedade, para apresentar os seus valores ao utilizador como alternativas no diálogo controlado - utilizando o maior valor da razão do ganho de informação e a informação do valor da propriedade ser não numérica; de acordo com o Algoritmo 1, resta-nos construir o conjunto A que tem como elementos os valores da melhor propriedade e apresentá-lo ao utilizador, para que este possa esclarecer o sistema sobre as suas pretensões.

Para além de especificar uma das possíveis alternativas apresentadas pelo sistema, o utilizador tem ainda três possibilidades para interagir como o sistema, nomeadamente: o símbolo `?` que significa "I do not know"; o símbolo `!` que significa "show all answers"; e o termo `quit`, que termina com o processo. No primeiro caso, o sistema apresenta um novo conjunto de alternativas de acordo com a avaliação das propriedades. No segundo caso, o sistema apresenta todas as soluções e o processo será terminado. No terceiro caso, o sistema simplesmente termina o processo.

Exemplo 5.3.5 Continuando o exemplo da questão 5.1 que tem vindo a ser apresentado, no passo anterior determinámos que a propriedade `http://purl.org/dc/terms/subject` é a melhor, para utilizar no diálogo com o utilizador, i.e., que satisfaz a Definição 5.3.5 - que tem os seus valores expressos em texto e o valor da sua razão de ganho de informação é o máximo relativamente às outras propriedades. Considere-se T_1 o conjunto que contém as ocorrências de T que correspondem à propriedade `http://purl.org/dc/terms/subject`.

O conjunto das alternativas A é formado pelos diferentes valores de todos os elementos do conjunto T_1. Do conjunto A são retiradas as alternativas que estão associadas diretamente com o termo do referente que originou a escolha desta propriedade e, por conseguinte, são retirados de T_1 os elementos que lhe correspondem. Neste caso, o termo é "Taj Mahal" e existe um triplo cujo valor em texto é exatamente o termo referido, nomeadamente

```
<http://dbpedia.org/resource/Taj_Mahal>
     <http://purl.org/dc/terms/subject>
          <http://dbpedia.org/resource/Category:Taj_Mahal> .
```

Por conseguinte, o Controlador do Discurso está em condições de iniciar um diálogo controlado com o utilizador e apresentar o conjunto de alternativas A, cuja resposta vai permitir esclarecer sobre que assunto está a referir-se. Assim, o Controlador do Discurso inicia o seguinte diálogo com o utilizador:

```
USER: "Where is Taj Mahal?"
SYSTEM: "Taj Mahal" refers to a:
```

```
1 - Buildings and structures completed in 1654
2 - Buildings and structures in Agra
3 - Mausoleums in India
4 - Mughal_architecture
5 - Domes
6 - Monuments and memorials in Uttar Pradesh
7 - Persian gardens in India
8 - Agra
9 - Indian architecture
10 - Iranian architecture
11 - Islamic architecture
12 - Mughal funary gardens in India
13 - Visitor attractions in Agra
14 - Visitor attractions in Uttar Pradesh
15 - World Heritage Sites in India
16 - Hotels established in 1903
17 - Hotels in Mumbai
18 - Taj Hotels, Resorts and Palaces
29 - Buildings and structures in Bhopal
20 - Palaces in Madhya Pradesh
21 - Hotels in Delhi
USER: 3
```

A alternativa escolhida pelo utilizador corresponde apenas ao triplo do conjunto T_1

```
<http://dbpedia.org/resource/Taj_Mahal>
     <http://purl.org/dc/terms/subject>
          <http://dbpedia.org/resource/Category:Mausoleums_in_India> .
```

Desta forma, o referente `Y` que esteve na origem da escolha da melhor propriedade fica restringido ao valor `http://dbpedia.org/resource/Taj_Mahal` (inicialmente eram 4 valores, ver Tabela 5.2). Por conseguinte, a escolha do utilizador restringiu consideravelmente o conjunto de soluções e orientou o Controlador do Discurso a ficar apenas com uma solução. Ou seja, a ambiguidade ficou esclarecida e o Controlador do Discurso está apto para processar a resposta e retorná-la ao utilizador. A solução encontrada é então

```
X = 'Agra, India'
Y = http://dbpedia.org/resource/Taj_Mahal
Z = 'Agra, India'
```

A resposta a apresentar ao utilizador, para além de objetiva e direta, irá informar sobre as entidades que serviram de suporte. Assim, são apresentadas as entidades da solução e respetivos resumos descritivos, disponibilizados pela DBpedia.

```
LOCATION/PLACE:
   Agra, India
RESOURCE:
   http://dbpedia.org/resource/Taj_Mahal
   The Taj Mahal is a mausoleum located in Agra, India.
```

Qualquer alternativa, excetuando a 18, conduzem o Controlador a uma única solução, logo a ambiguidade é esclarecida e a solução é apresentada ao utilizador.

Suponhamos que o utilizador escolhia a alternativa `18 - Taj Hotels, Resorts and Palaces`. O conjunto T_1 tem os seguintes triplos que corresponde a esta escolha

```
<http://dbpedia.org/resource/Taj_Mahal_Palace_&_Tower>
 <http://purl.org/dc/terms/subject>
  <http://dbpedia.org/resource/Category:Taj_Hotels,_Resorts_and_Palaces> .

<http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)>
 <http://purl.org/dc/terms/subject>
  <http://dbpedia.org/resource/Category:Taj_Hotels,_Resorts_and_Palaces> .
```

Esta escolha do utilizador é acrescentada ao contexto da questão, i.e., o conjunto T_1 vai ser restringido apenas a estas duas informações e a propriedade `http://purl.org/dc/terms/subject` não vai voltar a ser utilizada no processo de avaliação de propriedades.

De acordo com o Algoritmo 1, o próximo passo é restringir o conjunto de soluções S à escolha do utilizador. Assim, o conjunto de soluções fica restringido a duas, continuando a existir ambiguidade. Pelo que, voltamos ao passo "2:" do Algoritmo 1 - "Para cada referente colecionar as suas propriedades". Mais concretamente, para o novo conjunto de soluções voltar a repetir todos os passos apresentados nesta secção para determinar qual a melhor propriedade de forma a apresentar ao utilizador um conjunto de alternativas que poderão permitir esclarecer a ambiguidade.

Da mesma forma que ao exposto na primeira aplicação do Algoritmo 1, excluímos o referente `X` que está associado ao advérbio da questão, pois trata-se de um referente que está semanticamente relacionada com o que o utilizador pretende saber. Por analogia, o referente `Z` também é excluído porque a condição `is(X,Z)` torna-o semanticamente igual ao referente `X`. Desta forma, resta-nos o referente `Y` que está associado ao nome "Taj Mahal". O referente Y apenas contém os valores `http://dbpedia.org/resource/Taj_Mahal_Palace_&_Tower` e `http://dbpedia.org/resource/Taj_Mahal_Hotel_(Delhi)`, portanto o conjunto T é constituído apenas pelas propriedades associadas a estes valores. Ao todo são apenas 66 elementos que formam o conjunto T, associados a apenas 34 propriedades distintas. Temos que

$$Entropia(T) = 0.865965.$$

Propriedade	Entropia	Ganho de Informação	Razão do Ganho
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	0,869565	-0.00359987	-0.00413985
http://dbpedia.org/property/location	0	1,94195	ND
http://dbpedia.org/ontology/location	0	1,94195	ND
http://dbpedia.org/property/wikiPageUsesTemplate	0	1,94195	ND
http://dbpedia.org/property/latns	0	1,94195	ND
http://dbpedia.org/property/longd	0	1,94195	ND
http://dbpedia.org/property/longew	0	1,94195	ND
http://www.w3.org/2002/07/owl#sameAs	0	1,94195	ND
http://dbpedia.org/ontology/abstract	0	1,94195	ND
http://dbpedia.org/ontology/numberOfRooms	0	1,94195	ND
...	...	...	...
http://dbpedia.org/property/numberOfSuites	0	1,94195	ND
http://dbpedia.org/ontology/numberOfSuites	0	1,94195	ND
http://dbpedia.org/property/latd	0	1,94195	ND

Tabela 5.5: Resultados obtidos relativamente à Entropia, Ganho de Informação e Razão do Ganho de Informação das propriedades utilizadas para a clarificação da multiplicidade de soluções da questão "Where is the Taj Mahal?" - Segunda iteração da aplicação do Algoritmo 1.

A Tabela 5.5 apresenta os resultados obtidos na Entropia, no Ganho de Informação e na Razão do Ganho de Informação de algumas propriedades. Das 34 propriedades apenas uma tem entropia diferente de zero, a propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`, portanto a melhor propriedade. Mesmo que o seu valor seja numérico os seus valores terão de ser apresentados ao utilizador, uma vez que não existem mais alternativas de clarificação. O facto desta propriedade ter um ganho de informação negativo significa que existem muitos (senão todos os) valores comuns entre as diferentes soluções, podendo ter que ser necessário utilizar uma (ou várias) das propriedades que têm entropia igual a zero, para tentar obter uma clarificação do utilizador.

Assim, o conjunto das alternativas A é formado pelos diferentes valores de todos os elementos do conjunto de T_1, conjunto que contém as ocorrências de T que correspondem à propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`.

Do conjunto A são retiradas as alternativas que estão associadas diretamente com o termo do referente, que originou a escolha desta propriedade, e, por conseguinte, as respetivas ocorrências são retiradas do conjunto T_1. Continuando o diálogo com o utilizador, são apresentadas as seguintes alternativas

```
USER: "Where is Taj Mahal?"
SYSTEM: "Taj Mahal" refers to a:
1 - Buildings and structures completed in 1654
2 - Buildings and structures in Agra
...
21 - Hotels in Delhi
USER: 3
SYSTEM: "Taj Mahal" refers to a:
```

```
1 - Hotels in Delhi
2 - Taj Hotels, Resorts and Palaces
3 - Yago Geo Entity
4 - Architectural Structure
5 - Building
6 - Hotel
7 - Place
8 - Feature
9 - Thing
10 - Hotels in Mumbai
```

Apenas as alternativas `1 - Hotels in Delhi` e `10 - Hotels in Mumbai` esclarecem a ambiguidade, orientando o Controlador do Discurso para uma única solução. Caso o utilizador escolha uma das outras alternativas, o sistema tem de utilizar as propriedades que têm entropia igual a zero e ir apresentando os seus valores até o utilizador conseguir clarificar o Controlador do Discurso, ou o utilizador indicar que pretende ver todas as soluções.

Suponhamos agora que o utilizador aquando da apresentação do primeiro conjunto de alternativas não sabia indicar as suas pretensões, então utiliza o símbolo `?` para indicar que não sabe responder.

O Controlador do sistema escolhe a próxima melhor propriedade, para apresentar os seus valores como alternativas. De acordo com a Tabela 5.4 as próximas propriedades a escolher ou são numéricas ou o seus valores estão relacionados com os termos da questão (tais como, o termo `location`). Pelo que, a propriedade que resulta desta análise é a propriedade `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` e que tem um comportamento já explicado atrás.

O processo de diálogo termina quando o sistema consegue fornecer uma resposta ao utilizador, ou quando o utilizador indicar que pretende terminar e para tal utilizar o termo `quit`, ou que pretende ver todas as soluções, introduzindo o símbolo `!`. ▲

A interação, na forma de diálogo controlado, é usada não só para esclarecer os problemas de ambiguidade, mas também para ajudar a encontrar o caminho correto para a resposta pretendida. A cooperação entre o sistema e o utilizador torna-o capaz de chegar mais perto da resposta desejada pelo utilizador. Em muitos casos, o utilizador é o único que pode ajudar o sistema na dedução e interpretação da informação.

5.4 Conclusão

Os sistemas de Pergunta-Resposta para as línguas naturais na Web Semântica requerem mecanismos capazes de interpretar para além da pesquisa de palavras nos documentos

e, preferencialmente, encontrar a resposta correta, sem requerer a ajuda dos utilizadores para interpretar os documentos retornados. Neste capítulo, apresentámos um Controlador do Discurso que, através da análise da questão e do tipo de resposta esperada, permite fornecer respostas precisas às questões colocadas pelo utilizador na língua natural inglesa. O Controlador do Discurso representa a semântica das questões e a estrutura do discurso, que contempla as intenções do utilizador e o contexto das questões, o que permite adicionar a capacidade de lidar com múltiplas respostas. O Controlador do Discurso apresentado faz uso da interação, em forma de diálogo controlado, para clarificar problemas de ambiguidade e ajudar a encontrar o caminho correto, para a informação que fornece a resposta esperada.

Capítulo 6

Implementação do Sistema Proposto

Neste capítulo começamos por apresentar a base de conhecimento que constitui a fonte de conhecimento para o trabalho em causa. Depois, apresentamos algumas técnicas e respetivos predicados Prolog desenvolvidos para implementar os diferentes módulos, que constituem o sistema cooperativo de Pergunta-Resposta para a Web Semântica proposto. Na Secção 6.1, fazemos uma pequena introdução ao presente capítulo. Na Secção 6.2, apresentamos a base de conhecimento. Na restantes secções, apresentamos algumas técnicas e respetivos predicados Prolog implementados nos diferentes módulos do sistema. Finalmente, na Secção 6.10, estabelecemos as conclusões finais.

6.1 Introdução

O trabalho apresentado ao longo dos capítulos anteriores fica mais completo com uma implementação, que conduza a uma experimentação, avaliação e análise dos resultados obtidos. Ao longo deste capítulo, vamos apresentar as diferentes visões e estado atual de cada módulo, no que respeita à sua implementação. Apresentamos a base de conhecimento que está a ser utilizada atualmente, algumas técnicas e abordagens feitas em cada módulo do sistema, bem como os predicados implementados e que dão corpo ao sistema de Pergunta-Resposta que nos propusemos desenvolver. As definições dos predicados implementados e referidos ao longo deste capítulo são apresentados mais detalhadamente no Apêndice A.

6.2 Base de Conhecimento

A base de conhecimento é constituída, atualmente, pela ontologia OWL2 da DBpedia, pela base de dados DBpedia de acesso remoto e por uma pequena base de dados da DBpedia de acesso local, constituída por um conjunto de ficheiros com extensão `.nt`, que contém os triplos RDF e que estão armazenados num servidor local.

6.2.1 DBpedia

6.2.1.1 Acesso à Base de Dados DBpedia

As bases de dados DBpedia podem ser importados utilizando aplicações próprias ou acedida online utilizando uma variedade de interfaces. Para otimizar a pesquisa e o tempo de acesso à base de dados DBpedia, instalámos um servidor local Virtuoso[1] para armazenar parte da base de dados DBpedia. O servidor Virtuoso[2] é um servidor de dados que permite consultar e interrogar dados RDF. Atualmente fazem parte da base de dados no servidor local Virtuoso os seguintes dados: *Ontology Infobox Types*, cujo ficheiro de triplos RDF é `instance_types_en.nt`; *Titles*, cujo ficheiro de triplos RDF é `labels_en.nt`; *Ontology Infobox Properties*, cujo ficheiro de triplos RDF é `mappingbased_properties_en.nt`; *Persondata*, cujo ficheiro de triplos RDF é `persondata_en.nt`; *Short Abstracts*, cujo ficheiro de triplos RDF é `short_abstracts_en.nt`.

A base de dados DBpedia pode ser acedida online utilizando duas poderosas ferramentas:

- SPARQL[3] *endpoints*;
- o serviço de pesquisa online da DBpedia, *Lookup Service*[4].

A base de dados DBpedia instalada no servidor local Virtuoso pode ser acedida utilizando a ferramenta SPARQL.

SPARQL *Endpoint*

O SPARQL *endpoint* é uma ferramenta que nos permite questionar a base de dados DBpedia RDF[5] e foi apresentada na Secção 2.6.

A biblioteca Prolog *Semantic Web (semweb)*[6], do SWI-Prolog, permite manipular, consultar e questionar diretamente bases de dados RDF. O módulo `sparql_client.pl` fornece

[1] `http://joernhees.de/blog/2010/10/31/setting-up-a-local-dbpedia-mirror-with-virtuoso/`
[2] `http://virtuoso.openlinksw.com/`
[3] `http://www.w3.org/TR/rdf-sparql-query/`
[4] `http://wiki.dbpedia.org/lookup/`
[5] `http://www.w3.org/TR/REC-rdf-syntax/`
[6] `http://www.swi-prolog.org/pldoc/package/semweb.html`

um cliente SPARQL que permite lidar com a base de dados DBpedia, tanto no servidor local, como no servidor online. O predicado Prolog que nos permite consultar as bases de dados RDF é:

`sparql_query(+Query, -Result, +Options)`

Executa uma consulta SPARQL num HTTP SPARQL *endpoint*.

`Query` é um átomo e representa a consulta.

`Result` é um termo `rdf(S,P,O)` para os *queries* "CONSTRUCT" e "DESCRIBE", `row(...)` para o *query* "SELECT" e `true` ou `false` para o *query* "ASK".

`Options` são: `host(+Host), port(+Port), path(+Path)`, para estabelecer a localização do servidor; `search(+ListOfParams)`, fornece parâmetros adicionais tais como um conjunto de declarações RDF; `variable_names(-ListOfNames)`, unifica `ListOfNames` com uma lista de átomos que descrevem os nomes das variáveis no *query* "SELECT".

Exemplo 6.2.1 Suponhamos que pretendemos saber quais as entidades da DBpedia que são pessoas, i.e., as entidades que são classificadas como sendo do tipo "Person". A consulta à base de dados DBpedia disponível online, utilizando o questionário "SELECT", é feita da seguinte forma:

```
?- sparql_query('SELECT ?instancia WHERE { ?instancia <http://www.w3.
    org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/
    Person> }', Row, [ host('dbpedia.org'), path('/sparql/')]).
Row = row('http://dbpedia.org/resource/%C3%81kos_R%C3%A1thonyi') ;
Row = row('http://dbpedia.org/resource/%C3%81lvaro_Arz%C3%BA') ;
Row = row('http://dbpedia.org/resource/%C3%81lvaro_Colom') ;
Row = row('http://dbpedia.org/resource/%C3%81lvaro_Rafael_Gonz%C3%A1lez
    ') ;
Row = row('http://dbpedia.org/resource/%C3%81lvaro_de_Castro') ;
Row = row('http://dbpedia.org/resource/%C3%81ngel_Berlanga') ;
Row = row('http://dbpedia.org/resource/%C3%81ngel_Mar%C3%ADa_Villar') ;
Row = row('http://dbpedia.org/resource/%C3%81ngel_Suqu%C3%
    ADa_Goicoechea') .
```

Questionar o servidor local corresponde fazer uma consulta idêntica à anterior, mas onde o endereço e porto do servidor são os atribuídos localmente:

```
?- sparql_query('SELECT ?instancia WHERE { ?instancia <http://www.w3.
    org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/
    Person> }', Row, [ host('localhost'), port(8890), path('/sparql/')
    ]).
Row = row('http://dbpedia.org/resource/Aristotle') ;
Row = row('http://dbpedia.org/resource/Abraham_Lincoln') ;
Row = row('http://dbpedia.org/resource/Alain_Connes') ;
Row = row('http://dbpedia.org/resource/Allan_Dwan') .
```

Os resultados apresentados de ambas as consultas são diferentes, porque as bases de dados são diferentes em tamanho e ordem. ▲

Lookup Service

O serviço de pesquisa online da DBpedia, *Lookup Service*[7] permite encontrar URIs[8] relacionadas com uma determinada palavra-chave. Relacionada significa que tanto o rótulo de um recurso coincide, como um texto que é utilizado frequentemente na Wikipédia para se referir a um recurso específico coincide (por exemplo, o recurso `http://dbpedia.org/resource/United_States` pode ser procurado através da string "USA"). Os resultados são ordenados utilizando o número de ligações apontadas por outras páginas da Wikipédia e apresentados numa página de resultados.

A utilização deste serviço pode ser feita de duas formas e através de dois interfaces distintos: pesquisar por palavra-chave e pesquisar através do prefixo de um termo. A URL dos interfaces de pesquisa tem a forma

```
http://lookup.dbpedia.org/api/search.asmx/<API>?<parâmetros>
```

Pesquisar palavra-chave - A pesquisa feita através de uma palavra-chave é utilizada para encontrar recursos da DBpedia relacionados com uma determinada *string* (cadeia de caracteres). A *string* pode representar uma palavra simples ou múltiplas palavras. No interface de pesquisa, o termo <API> é substituído por *keywordSearch*.

Por exemplo, imagine que pretendemos os locais relacionados com a palavra "lisboa", a URL tem a seguinte forma `http://lookup.dbpedia.org/api/search.asmx/KeywordSearch?QueryClass=place&QueryString=lisboa`.

Pesquisa prefixo (ou seja, Auto-completar) - A pesquisa feita através de um prefixo é utilizada para implementar uma forma automática de completar a informação de entrada. Para uma determinada palavra-chave parcial como "Lisb" o correspondente interface retorna as URI's dos recursos da DBpedia que lhe estão relacionadas, tais como `http://dbpedia.org/resource/Lisbon`. No interface de pesquisa, o termo <API> é substituído por *PrefixSearch*.

Por exemplo, imagine que pretendemos os 5 primeiros recursos da DBpedia cujo rótulo começa com o termo "lisb", a URL tem a seguinte forma

```
http://lookup.dbpedia.org/api/search.asmx/PrefixSearch?QueryClass=
&MaxHits=5&QueryString=lisb
```

[7] `http://wiki.dbpedia.org/lookup/`
[8] `http://tools.ietf.org/html/rfc3986/`

Os parâmetros que constituem a URL do interface de pesquisa são:

QueryString - uma *string* para que um recurso URI DBpedia deve ser encontrado.

QueryClass - uma classe da Ontologia DBpedia a que os resultados devem pertencer (para recursos sem tipo ou que pertençam a "Thing", este parâmetro permanece vazio).

MaxHits - número máximo de resultados retornados (predefinido: 5)

Os resultados da pesquisa são devolvidos em formato XML e para cada resultado é devolvido o conjunto de informação pela seguinte ordem: URI; Rótulo; (Short) Descrição; Classes (URI e rótulo); Categorias (URI e rótulo); Refcount (número de inlinks da página Wikipedia). Para compatibilidade com versões anteriores, também são devolvidas as informações <Templates vazio/> e <Redirects etiquetas/>.

A biblioteca do Swi-Prolog disponibiliza predicados que nos permitem trabalhar diretamente com estruturas URL e XML, tais como: `http_open/3`[9], `load_structure/3`[10], `close/1`[11]. Para ilustrar a forma como estes predicados são utilizados considere o seguinte exemplo:

Exemplo 6.2.2 Suponhamos que pretendemos os 5 primeiros recursos da DBpedia que está relacionado o termo "lisboa". Em Prolog é possível construir um conjunto de instruções que permitem fazer a ligação à base de dados DBpedia, utilizando a ferramenta de pesquisa da DBpedia e obter os resultados pretendidos. Assim,

```
http_open('http://lookup.dbpedia.org/api/search.asmx/KeywordSearch?
    QueryClass=&MaxHits=5&QueryString=lisboa', In, []),
load_structure(In, [element('ArrayOfResult', _, A)], []),
close(In),
member(element('Result', [], B), A),
member(element('URI', [], [URIName]), B).
```

A variável `In` é a stream que representa o resultado da pesquisa efetuada pela ferramenta de pesquisa à base de dados DBpedia. A variável `A` representa cada um dos conjuntos de informação associada a cada recurso da DBPedia que está na stream `In`. A `URIName` é o recurso da DBpedia que procuramos. ▲

Genericamente, quando pretendemos encontrar os recursos da DBpedia que estejam relacionados com um determinado "Termo", utilizando os predicados e as instruções apresentadas anteriormente, podemos utilizar um dos seguintes predicados Prolog:

[9] `http://www.swi-prolog.org/pldoc/man?predicate=http_open/3`
[10] `http://www.swi-prolog.org/pldoc/man?predicate=load_structure/3`
[11] `http://www.swi-prolog.org/pldoc/man?predicate=close/1`

`resources(-URIList, +Name)`

`URIList` é a lista de resultados, com a informação associada a cada recurso da DBpedia, que foi devolvido pela ferramenta de pesquisa, e que estão relacionados com o termo `Name`.

`name(-URI, +Name)`

`URI` é a URI do recurso da DBpedia que está relacionado com o termo `Name`.

6.2.1.2 Ontologia DBpedia

A ontologia DBpedia[12] é uma ontologia superficial e de domínio cruzado, que foi criada manualmente baseando-se na forma mais usual do uso das *infoboxes* dentro da Wikipédia. Este facto faz com que a sua estrutura semântica contenha algumas falhas de consistência lógica e apresente lacunas semânticas significativas na sua hierarquia.

A ontologia cobre atualmente mais de 359 classes que formam uma hierarquia de subsunção e são descritos por 1.775 propriedades diferentes. A versão atual da ontologia DBpedia contém cerca de 2.350.000 instâncias: 573.000 locais; 764.000 pessoas; 333.000 empregos; 202.000 espécies; 192.000 organizações.

A comunidade DBpedia fornece uma versão OWL da ontologia para download[13], que à data de escrita deste trabalho se encontra na versão 3.8.

Thea[14] [107] é uma biblioteca Prolog que fornece um suporte completo para consultar e processar ontologias OWL2, diretamente de dentro de programas Prolog. *Thea* utiliza uma biblioteca RDF do Prolog para análise e serialização de ontologias, onde o modelo principal é independente da RDF e é baseado no estilo da sintaxe funcional OWL2. Este facto, permite a manipulação direta dos axiomas da ontologia através da base de dados Prolog, em vez de utilizar indiretamente os triplos RDF. A versão atual, *Thea2*, para além de fornecer os predicados que suportam as entidades básicas OWL (tais como, classes, propriedades e indivíduos), foi redesenhada para suportar a linguagem OWL2 seguindo exatamente a sua especificação da sintaxe estrutural: todos os axiomas da ontologia correspondem a factos na base de dados Prolog. Isto permite consultar a ontologia através da consulta direta da base de dados Prolog usando predicados com variáveis como argumentos.

Assim, para converter a ontologia DBpedia OWL2 para predicados Prolog utilizamos o predicado Prolog `load_axioms/3`[15], contido no ficheiro `owl2_io.pl` da biblioteca *Thea*[16]:

`load_axioms(+File, +Fmt, +Opts)`

Converte os axiomas do modelo OWL2 contidos no ficheiro `File` para Prolog.

[12] `http://wiki.dbpedia.org/Ontology?v=181z`
[13] `http://wiki.dbpedia.org/Downloads38#dbpedia-ontology`
[14] `http://www.semanticweb.gr/thea/index.html`
[15] `http://www.berkeleybop.org/blipdoc/doc_for?object=owl2_io%3aload_axioms%2f3`
[16] `http://www.berkeleybop.org/blipdoc/doc/users/cjm/pllib/thea2/index.html`

`Fmt = owl | owlx | prolog | plsyn | dlp | owlapi(_) | ....`

`Opts` são especificações `Fmt`.

Os ficheiros Prolog que representam a biblioteca *Thea* são incluídos no trabalho utilizando o predicado Prolog `use_module/1`[17].

Mais concretamente, a biblioteca *Thea* é incluída e traduzida para Prolog através dos seguintes comandos:

```
:- use_module(library(thea2/owl2_io)).
:- use_module(library(thea2/owl2_model)).
:- load_axioms('dbpedia_3.8.owl',owl).
```

Os predicados Prolog, tais como `class/1`, `subClassOf/2`, `subOjectPropertyOf/2` e `subDataPropertyOf/2` (entre outros), podem ser utilizados para interrogar a ontologia.

6.2.2 WordNet

A base de dados WordNet é necessária no módulo Contexto Ontológico, onde é feita a extensão aos sinónimos, hiperónimos e/ou hipónimos dos termos que constituem a representação semântica da questão DRS e cuja procura por recursos da base de conhecimento não é bem sucedida. Esta extensão é feita com o objetivo de aumentar o conjunto dos termos de pesquisa e que permita melhorar os resultados da procura de informação na base de conhecimento. Para utilizar a base de dados Prolog da DBpedia, desenvolvemos alguns predicados que nos permitem obter a informação pretendida, tais como os sinónimos:

`synonymous(+Word, Category, -Synonymous)`

> `Synonymous` é a palavra ou termo da base de dados da DBpedia que tem uma relação de sinonímia com a palavra `Word`. `Category` representa o grupo *synset* ao qual `Word` pertence, pode tomar os valores *n(oun), v(erb), a(djective), s(atellite adjective), r(adverb).*

Atualmente, apenas os sinónimos diretos (que se encontram num primeiro nível) estão acessíveis. Como trabalho futuro, pretendemos acrescentar predicados que nos permitam aceder a outros níveis da informação, desde sinónimos, relações que se podem estabelecer entre dois *synsets* e até informação que nos permita melhorar a interpretação da questão (através de regras de inferência, raciocínio sobre o conhecimento).

Os ficheiros Prolog que representam a base de dados WordNet são incluídos no trabalho utilizando o predicado Prolog `include/1`[18].

[17] `http://www.swi-prolog.org/pldoc/man?predicate=use_module%2F1`
[18] `http://www.swi-prolog.org/pldoc/man?predicate=include/1`

6.3 Análise Sintática e Interpretação Semântica da Questão

A Análise Sintática e Interpretação Semântica da questão, como apresentado respetivamente nas Secções 4.2 e 4.3, traduzem a fase do sistema em que a questão, expressa em Inglês e introduzida pelo utilizador, é transformada na sua correspondente representação semântica DRS. Primeiro, é feita a análise sintática, utilizando estruturas gramaticais da língua natural Inglesa. Depois, é feita a interpretação semântica da estrutura sintática, através da identificação de algumas marcações na estrutura sintática e correspondente transformação semântica, utilizando algumas das regras apresentadas na Secção 4.3.

A estrutura sintática da questão é obtida recorrendo ao interpretador *Stanford*[19].

O conjunto de regras que permite identificar algumas marcações na estrutura sintática da questão e reescrevê-las com um significado semântico, não está completo. Uma vez que este não é o principal foco do trabalho desenvolvido, apenas foram estabelecidas algumas regras que permitem validar e verificar os exemplos apresentados ao longo da tese e os exemplos utilizados na avaliação do sistema.

Embora o resultado deste processo seja fundamental para a execução dos restantes módulos do sistema, a sua implementação é uma tarefa bastante difícil e demorada. O tempo, o esforço e a complexidade que uma tarefa desta natureza exige, i.e., implementar uma ferramenta que transforme uma questão expressa em língua natural na correspondente representação semântica, quase que daria uma nova tese. Pelo que, o processo de conversão de uma questão expressa em língua natural na sua representação semântica é feito, ainda, de forma manual.

A análise e atribuição do tipo de questão e resposta esperada também é feita, nesta fase do sistema e por enquanto, manualmente. Como referido na Secção 4.3.1, a classificação é feita de acordo com a correspondência apresentada na Tabela 6.1.

Como falámos anteriormente, o Prolog é a linguagem de programação utilizada para a implementação do sistema cooperativo de Pergunta-Resposta proposto. Pelo que a DRS de uma questão, e de acordo com a definição 3.8.2, tem o seguinte formato em Prolog:

```
drs(Tipo_Questão,
    Lista_Referentes,
    Lista_Pressupostos,
    Lista_Predicados_Principais)
```

onde

`Tipo_Questão` - é a identificação do tipo de questão;

`Lista_Referentes` - é a lista de referentes existenciais e universais associados às entidades da questão e que surgem da interpretação semântica da questão;

[19] `http://nlp.stanford.edu/software/lex-parser.shtml`

Marcação	Tipo de questão e resposta esperada
yes/no	questões de polaridade
PERSON/ORGANIZATION	"who" - questão sobre uma pessoa ou uma organização
PLACE/LOCATION	"where" - questão sobre um local ou uma localização
DATE/TIME	"when" - questão sobre uma data ou uma hora
THING/OBJECT	"what" - questão sobre uma coisa ou um objeto
CHOICE	"which" - questão sobre uma escolha, algo que verifique uma dada condição

Tabela 6.1: Informação sobre atribuição feita às questões de acordo com o seu tipo e resposta esperada.

`Lista_Pressupostos` - é a lista de predicados pressupostos da questão e que condicionam as entidades da questão e estão diretamente relacionados com os termos linguísticos da questão;

`Lista_Predicados_Principais` - é a lista de predicados principais da questão e que têm de ser verificados pelas entidades da questão que validam os predicados pressupostos. Os predicados principais são obtidos da interpretação semântica da questão.

Exemplo 6.3.1 Para ilustrar como procedemos nesta fase do sistema, embora de forma manual, considere a questão já utilizada na Secção 4.3

```
Who is Barbara Jordan?
```
(6.1)

pertencente aos conjunto de questões apresentado no TREC 9 (*The Ninth Text REtrieval Conference* [108]). Utilizando o analisador Stanford, obtemos a seguinte informação relativa à análise sintática da questão:

Tagging

```
Who/WP is/VBZ Barbara/NNP Jordan/NNP ?/.
```

Parse

```
(ROOT
  (SBARQ
    (WHNP (WP Who))
    (SQ (VBZ is)
      (NP (NNP Barbara) (NNP Jordan)))
    (. ?)))
```

Typed dependencies

```
attr(is-2, Who-1)
root(ROOT-0, is-2)
nn(Jordan-4, Barbara-3)
nsubj(is-2, Jordan-4)
Typed dependencies, collapsed
attr(is-2, Who-1)
root(ROOT-0, is-2)
nn(Jordan-4, Barbara-3)
nsubj(is-2, Jordan-4)
```

Statistics

Tokens: 5

De acordo com a classificação apresentada na Tabela 6.1, a questão em causa tem como tipo e resposta esperada a marcação "PERSON/ORGANIZATION". As questões que são classificadas com duas marcações relativas ao tipo de questão e resposta esperada, originam duas interpretações distintas e consequentemente predicados pressupostos correspondentes, também distintos. Para o exemplo apresentado, significa que podemos ter duas interpretações: o nome "Barbara Jordan" refere-se uma pessoa ou a uma organização.

Na Secção 4.3 apresentámos um conjunto de regras que permitem transformar a estrutura sintática de uma questão na sua representação semântica e utilizámos também este exemplo para ilustrar a transformação. Relembrando: o pronome `who` gera um novo referente, por exemplo `X`, com adição do prefixo `who`, e a condição `person(X)` (ou a condição `organization(X)`); os nomes "Barbara" e "Jordan" estão no mesmo agrupamento da estrutura sintática, pelo que é gerado um novo referente, por exemplo `Y`, e a condição `name(Y, 'Barbara Jordan')`; o verbo `is` é utilizado para definir a condição `is(X,Y)`, onde os argumentos são os referentes já criados e que estão relacionados no discurso. Na nova representação da questão, a representação semântica, os referentes e as condições criadas substituem as correspondentes na estrutura sintática. Assim, a interpretação semântica é expressa pelas seguintes representações DRS:

```
drs('PERSON/ORGANIZATION',
    [who–X, exist–Y],
    [name(Y, 'Barbara Jordan'), person(X)],
    [is(X,Y)]).

drs('PERSON/ORGANIZATION',
    [who–X, exist–Y],
    [name(Y, 'Barbara Jordan'), organization(X)],
    [is(X,Y)]).
```

Algoritmo 2 Contexto Ontológico

Entrada: $T = \{t \mid t$ é um termo da DRS da questão$\}$
Saída: Obter $\forall t \in T, Ontologia(t) = \{ontologia\_t \in Ontologia \mid ontologia\_t$ está relacionado com $t\}$
  Para cada termo $t \in T$ **faz**
    Enquanto não percorrer toda a base de conhecimento **faz**
      ontologia_t := entidade da ontologia que está relacionado com t
      adicionar *ontologia_t* ao conjunto $Ontologia(t)$
    Fim Enquanto
  Fim Para

A lista dos predicados pressupostos para a primeira DRS é `[name(Y, 'Barbara Jordan'), person(X)]`, e para a segunda DRS é `[name(Y, 'Barbara Jordan'), organization(X)]`; para ambas as DRS, a lista dos predicados principais é `[is(X,Y)]` e a lista de referentes é `[who-X, exist-Y]`, onde `X` é uma entidade universalmente quantificada, que corresponde à solução da questão, e `Y` uma entidade da base de conhecimento existencialmente quantificada. ▲

6.4 Contexto Ontológico

O módulo Contexto Ontológico, apresentado na Secção 4.4, tem como função fazer o mapeamento entre os termos linguísticos da questão e as entidades da ontologia. Ou seja, para cada termo correspondente a um predicado da DRS, o sistema tem de encontrar recursos da ontologia que lhe sejam semelhantes (ou estejam relacionados). Para tal, faz a verificação de semelhanças entre os rótulos, de acordo com as strings dos termos e recursos envolvidos, considerando abreviaturas, acrónimos, e o domínio do conhecimento. Para maximizar a pesquisa, o sistema procura pelas classes, propriedades ou instâncias da ontologia, cujos rótulos correspondam exata ou parcialmente a um termo da DRS. Assim, o Algoritmo 2 ilustra o procedimento efetuado pelo sistema aquando da execução do módulo do Contexto Ontológico, para mapeamento dos termos linguísticos nos termos da ontologia.

Nesta fase da implementação, o mapeamento é feito através da correspondência exata ou parcial entre o termo linguístico e o rótulo das entidades da ontologia.

Definição 6.4.1 *As entidades da ontologia classificam-se em* ***classes***[20]*,* ***recursos***, ***propriedades*** *e* ***dados****.*

1. *As classes são identificadas por uma referência URI da forma* `http://dbpedia.org/ontology/Name`*, onde* `Name` *corresponde ao nome da classe e destingue-se dos outros nomes por ser inicializado com letra maiúscula. Por exemplo, a URI* `http://dbpe`

[20] `http://mappings.dbpedia.org/server/ontology/classes/`

`dia.org/ontology/Country` identifica a entidade da ontologia que corresponde à classe "Country";

2. *Os recursos são identificados por uma referência URI da forma `http://dbpedia.org/resource/Name`, onde `Name` corresponde ao nome do recurso. Por exemplo, a URI `http://dbpedia.org/resource/Portugal` identifica a entidade da ontologia que corresponde ao recurso associado ao nome "Portugal";*

3. *As propriedades são identificadas por uma referência URI da forma `http://dbpedia.org/ontology/name`, onde `name` corresponde ao nome da propriedade, e constituído apenas por letras minúsculas. Por exemplo, a URI `http://dbpedia.org/ontology/language` identifica a entidade da ontologia que corresponde à propriedade "language". Existe ainda um conjunto de propriedades genéricas e que não estão associadas a qualquer classe, mas que poderão aparecer na definição de objetos (instâncias de classes). Estas propriedades são identificadas por uma referência URI da forma `http://dbpedia.org/property/name`, onde `name` corresponde ao nome da propriedade. Por exemplo, a URI `http://dbpe dia.org/property/city` identifica a entidade da DBpedia que corresponde à propriedade genérica "city";*

4. *Os dados podem ser classes, recursos, texto, valores numéricos ou ainda atributos XML Schema*[21].

As classes distinguem-se das propriedades por terem a sua identificação "Name" iniciada com letra maiúscula.

Genericamente, para encontrar as entidades da DBpedia que estejam relacionados com um determinado termo, respetivamente classes, propriedades e recursos, construímos os seguintes predicados Prolog:

`classOf(-Class, +Term)`

`Class` é a URI da classe que está relacionada com o termo linguístico `Term`.

`property(-Property, +Term)`

`Property` é a URI da propriedade genérica que está relacionada com o termo linguístico `Term`.

`name(-URIName, +Name)`

`URIName` é o recurso URI da DBpedia que está relacionado com o nome `Name`. Este predicado utiliza o serviço de pesquisa online da DBpedia, *Lookup Service*[22] para encontrar os recursos pretendidos.

[21] Atributos XML Schema são literais que identificam um determinado valor e seu tipo na classificação XML Schema. Por exemplo, `literal(type('http://www.w3.org/2001/XMLSchema#int',123))`.

[22] `http://wiki.dbpedia.org/lookup/`

Exemplo 6.4.1 Continuando o exemplo da questão 6.1, na DBpedia existem várias recursos que estão relacionados com o nome `Barbara Jordan` e a verificação do predicado `name(Y, 'Barbara Jordan')` resultam, por exemplo,

```
Y = 'http://dbpedia.org/resource/Barbara_Jordan'
Y = 'http://dbpedia.org/resource/Barbara_Jordan_(tennis)'
Y = 'http://dbpedia.org/resource/Barbara_Jordan_High_School'
Y = 'http://dbpedia.org/resource/Barbara_Jordan_Health_Policy_Scholars'
Y = 'http://dbpedia.org/resource/Barbara_Jordan_High_School_for_Careers'
Y = 'http://dbpedia.org/resource/Barbara_Jordan_(poet)'
```

O predicado `person(X)` não está implementado, pois o termo `person` pode ser um recurso, uma classe ou uma propriedade. Estes predicados são tratados como casos genéricos onde é detetado o termo e feita uma pesquisa na ontologia e na DBpedia de forma a encontrar entidades que estão associadas ao termo. Para tal, são utilizados os predicados `classOf/2`, `property/2` e `name/2`. Para o termo `person` existem, entre muitas, as seguintes URIs da DBpedia que lhe estão relacionadas:

```
?- classe(X,'person').
X = 'http://dbpedia.org/ontology/MilitaryPerson'
X = 'http://dbpedia.org/ontology/MilitaryPerson'
X = 'http://dbpedia.org/ontology/PersonFunction'
X = 'http://dbpedia.org/ontology/Person'

?- property(X,'person').
X = 'http://dbpedia.org/ontology/personName'
X = 'http://dbpedia.org/ontology/Person/height'
X = 'http://dbpedia.org/ontology/personFunction'
X = 'http://dbpedia.org/ontology/person'

?- name(X,'person').
X = 'http://dbpedia.org/resource/Model_(person)'
X = 'http://dbpedia.org/resource/Chinese_people'
X = 'http://dbpedia.org/resource/First-person_shooter'
X = 'http://dbpedia.org/resource/Black_people'
```

▲

6.5 Avaliação Semântica

O módulo Avaliação Semântica, apresentado na Secção 4.5, tem como função restringir a representação semântica inicial da questão, através da adição de novas condições que limitam as entidades do discurso associadas à questão.

Como vimos, esta fase caracteriza-se como sendo a Avaliação Pragmática do sistema, onde a representação semântica da questão é transformada num problema de satisfação de restrições. O problema de satisfação de restrições é constituído por um conjunto de regras de inferência, traduzidas em Prolog, que permite verificar a consistência das entidades da ontologia obtidas durante o mapeamento efetuado na fase do Contexto Ontológico.

Assim, os predicados da DRS são interpretados como simples factos Prolog, onde as restrições do problema são os predicados da representação semântica e as variáveis do problema são as entidades associadas aos referentes da DRS. As soluções do problema de satisfação de restrições representam o conjunto de entidades da ontologia que suportam a resposta à questão colocada pelo utilizador.

Mais concretamente o sistema tem de encontrar os valores para as variáveis da representação semântica da questão que são solução da questão. Ou seja, para as correspondências encontradas no Contexto Ontológico, o sistema tem de encontrar as entidades/recursos da ontologia que verificam os triplos RDF que correspondem às condições da representação semântica. As soluções e os respetivos triplos que verificam a estrutura semântica são adicionados ao contexto da questão.

Para este efeito, os factos Prolog da DRS são traduzidos em consultas SPARQL, que questionam a base de conhecimento, e as soluções destas consultas que verifiquem a DRS da questão são então entidades candidatas a solução da questão inicial.

Nesta fase, os seguintes predicados fazem parte dos predicados Prolog desenvolvidos:

`query(+X, +Property, -Y)`

> Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `Y` que verificam a consulta. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação direta.

`query(+X, +Property, +Term, -Y)`

> Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `Y` que verificam a consulta, quando estão relacionadas com o termo `Term`. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação direta.

`query(-X, +Property, +Term, -Y)`

> Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `X, Y` que verificam a consulta, quando as entidades `Y` estão relacionadas com o termo `Term`. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzido em triplos RDF. O triplo RDF é interpretado como relação direta.

`query(+X, -Property, +Term, -Y)`

Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `Property`, `Y` que verificam a consulta, quando as entidades `Y` estão relacionadas com o termo `Term`. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação direta.

`isa(-X,+Y)`

Verifica/estabelece a semelhança entre `X` e `Y`. No caso mais simples permite verificar ou estabelecer que `X = Y`.

`add_to_context(+Data)`

O facto `Data` é adicionado ao contexto do discurso da questão. Este predicado recorre ao predicado pré-definido `assert(+Term)` (a expressão `Term` é definida como a última cláusula ou facto do predicado).

Como a biblioteca Prolog já contém o predicado pré-definido (*built-in*) `is/2`, que permite atribuir/validar atribuições de valores numéricos a variáveis, e uma vez que pretendemos utilizá-lo na nossa implementação, o termo `is` na estrutura semântica corresponde ao predicado `isa/2`.

Exemplo 6.5.1 Voltando ao exemplo da questão 6.1, nesta fase, o sistema tem de encontrar os valores para as variáveis da representação semântica da questão que são solução da questão. Ou seja, para as correspondências encontradas no Contexto Ontológico, o sistema tem de encontrar as entidades/recursos da ontologia que verificam os triplos RDF que correspondem às condições da representação semântica.

Para os recursos Y que verificam a condição `name(Y,'Barbara Jordan')`, por exemplo, `Y = 'http://dbpedia.org/resource/Barbara_Jordan'`, ao validar a condição `isa(X,Y)`, torna as entidades iguais `X = Y = 'http://dbpedia.org/resource/Barbara_Jordan'`. Pelo que, a validação da condição `person(X) = person('http://dbpedia.org/resource/Barbara_Jordan')` é feita pelo triplo RDF da DBpedia:

```
<http://dbpedia.org/resource/Barbara_Jordan>
      <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
            <http://dbpedia.org/ontology/Person>
```

Desta forma, toda a informação necessária para a validação da estrutura semântica da questão é adicionada ao contexto da questão, nomeadamente: o mapeamento, os triplos RDF e a solução:

- O mapeamento na ontologia feito ao termo `person` e que verifica a estrutura semântica da questão é `http://dbpedia.org/ontology/Person`. Pelo que o seguinte facto é adicionado ao contexto da questão:

```
map('person', 'http://dbpedia.org/ontology/Person').
```

- O triplo que verifica a condição `person('http://dbpedia.org/resource/Barbara_Jordan')` é adicionado ao contexto utilizando o seguinte facto:

```
triple('http://dbpedia.org/resource/Barbara_Jordan',
       'http://www.w3.org/1999/02/22-rdf-syntax-ns#type',
       'http://dbpedia.org/ontology/Person').
```

- A solução é adicionada ao contexto da questão utilizando o seguinte facto

```
solution(['http://dbpedia.org/resource/Barbara\_Jordan',
          'http://dbpedia.org/resource/Barbara\_Jordan'],
         [map('person', 'http://dbpedia.org/ontology/Person')],
         [triple('http://dbpedia.org/resource/Barbara_Jordan',
                 'http://www.w3.org/1999/02/22-rdf-syntax-ns#type',
                 'http://dbpedia.org/ontology/Person')]).
```

▲

6.6 Controlador do Discurso

O Controlador do Discurso, apresentado na Secção 4.6, é a componente principal do sistema de Pergunta-Resposta subjacente a este trabalho. Este módulo tem como principal função controlar todos os passos desde que a questão em língua natural, colocada pelo utilizador, é transformada na sua representação semântica até ser possível fornecer uma resposta ao utilizador.

Embora o Controlador do Discurso contenha alguns passos ainda em fase de implementação, tais como o gestor de diálogos e o processamento da resposta, os seguintes predicados Prolog fazem parte dos predicados desenvolvidos nesta fase. São usados nas fases do Contexto Ontológico e Avaliação Semântica e que permitem determinar todas as soluções da questão inicial:

`presuppositions_solutions(+Variables, +Presuppositions_List, -Presuppositions_Solutions_List)`

Extrai a lista de entidades da ontologia, valores das variáveis, que verificam os predicados pressupostos que constituem a DRS da questão. `Varibles` é a lista de variáveis associada aos referentes da DRS da questão, `Presuppositions_List` é a lista de predicados pressupostos da DRS da questão e `Presuppositions_Solutions_List` é a lista de soluções, valores das variáveis, que verificam os pressupostos.

`main_solution(+Variables, +Presuppositions_Solutions_List, +Main_Predicates_List, -Main_Solutions_List)`

Extrai da lista de entidades da ontologia, que são solução dos predicados pressupostos, as entidades que verificam os predicados principais que constituem a DRS da questão. `Varibles` é a lista de variáveis associados aos referentes da DRS da questão, `Presuppositions_Solutions_List` é a lista de entidades que são solução dos predicados pressupostos da DRS da questão, `Main_Predicates_List` é a lista de predicados principais da DRS da questão e `Main_Solutions_List` é a lista de soluções, valores das variáveis, dos predicados pressupostos que verificam os predicados principais.

`get_solution(-Variables_Values, +Predicates_List)`

Extrai as soluções que verificam os predicados. `Variables_Values` é o valor das variáveis associados aos referentes da DRS da questão e `Predicates_List` é a lista de predicados que têm de ser verificados.

`process_list(+Predicates_List)`

Processa a lista de predicados `Predicates_List` que constituem a DRS da questão, i.e., invoca e controla a execução do Contexto Ontológico e a Avaliação Semântica.

`find(+Term, -X)` **e** `find(+Term,-X,-Y)`

Fazem o mapeamento dos predicados linguísticos da DRS da questão nos termos da ontologia, bem como a sua validação na ontologia. O predicado Prolog `find/2` corresponde aos predicados linguísticos com apenas um argumento `X`, onde `Term` corresponde ao nome do predicado. O predicado Prolog `find/3` corresponde aos predicados linguísticos com dois argumentos `X` e `Y`, onde `Term` corresponde ao nome do predicado.

`instance(-X, +Class)`

Verifica/determina se a entidade `X` é uma instância da classe da ontologia `Class`.

`classOf(-Class, +X)`

Determina/verifica se a classe `Class` está relacionada com o termo `X`. Se o termo `X` é um termo linguístico então a classe `Class` é o mapeamento na ontologia do termo. Se o termo `X` é uma entidade da ontologia então verifica/determina que o termo é uma instância da classe `Class`.

6.7 Gestor de Diálogos

O gestor de diálogos é invocado caso o conjunto de soluções obtidas gere ambiguidade tendo em conta o tipo de resposta a fornecer, por exemplo multiplicidade de respostas com contextos diferentes. O Controlador do Discurso ativa o mecanismo de clarificação de forma a desenvolver um diálogo controlado com o utilizador. O mecanismo de clarificação

segue o Algoritmo 1. Na Secção 5.3 apresentamos, com algum pormenor, como este mecanismo de clarificação funcionava e como através dele era possível clarificar o sistema sobre as pretensões do utilizador e por conseguinte fornecer a resposta à questão colocada pelo utilizador. Portanto, alguns pontos como o cálculo das entropias, dos ganhos de informação e razão do ganho de informação não serão expostos com pormenor para o exemplo que estamos a apresentar ao longo deste capítulo.

Exemplo 6.7.1 Voltando ao exemplo da questão 6.1, no passo anterior o sistema encontrou 3 soluções que verificam a estrutura semântica da questão:

```
solution(['http://dbpedia.org/resource/Barbara_Jordan',
          'http://dbpedia.org/resource/Barbara_Jordan'],
         [map(person, 'http://dbpedia.org/ontology/Person')],
         [trile('http://dbpedia.org/resource/Barbara_Jordan',
                'http://www.w3.org/1999/02/22-rdf-syntax-ns#type',
                'http://dbpedia.org/ontology/Person')]).

solution(['http://dbpedia.org/resource/Barbara_Jordan_(tennis)',
          'http://dbpedia.org/resource/Barbara_Jordan_(tennis)'],
         [map(person, 'http://xmlns.com/foaf/0.1/Person')],
         [triple('http://dbpedia.org/resource/Barbara_Jordan_(tennis)',
                 'http://www.w3.org/1999/02/22-rdf-syntax-ns#type',
                 'http://xmlns.com/foaf/0.1/Person')]).

solution(['http://dbpedia.org/resource/Barbara_Jordan_(poet)',
          'http://dbpedia.org/resource/Barbara_Jordan_(poet)'],
         [map(person, 'http://xmlns.com/foaf/0.1/Person')],
         [triple('http://dbpedia.org/resource/Barbara_Jordan_(poet)',
                 'http://www.w3.org/1999/02/22-rdf-syntax-ns#type',
                 'http://xmlns.com/foaf/0.1/Person')]).
```

Estamos na presença de multiplicidade de soluções para a questão, pelo que se gera uma ambiguidade e o sistema não tem informação suficiente para saber qual a resposta correta. Assim, é necessário invocar o mecanismo de clarificação das intenções do utilizador:

- No primeiro passo é necessário para cada referente colecionar as suas propriedades. No contexto da questão, embora o referente `X` esteja relacionado com o que o utilizador pretende saber, i.e., a resposta à questão, e a condição `is(X,Y)` tornar os referentes semanticamente iguais, como a ambiguidade está relacionada com este referente, não será excluído. Assim, é invocado o predicado `get_properties(+Referent_list_values)`, com `Referent_list_values = ['http://dbpedia.org/resource/Barbara_Jordan','http://dbpedia.org/resource/Barbara_Jordan_(tennis)','http:`

Propriedade	Entropia	Ganho de Informação	Razão do Ganho
http://purl.org/dc/terms/subject	0,0571429	1,14528	20,0424
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	0,371283	0,831139	2,23856
http://dbpedia.org/property/wikiPageUsesTemplate	0,594361	0,608061	1,02305
http://dbpedia.org/property/name	1,18872	0,0137004	0,0115253
http://xmlns.com/foaf/0.1/givenName	1,58496	-0,38254	-0,241356
http://xmlns.com/foaf/0.1/name	1,58496	-0,38254	-0,241356
http://xmlns.com/foaf/0.1/surname	1,58496	-0,38254	-0,241356
http://dbpedia.org/ontology/birthPlace	0	1,20242	NA
http://dbpedia.org/ontology/deathDate	0	1,20242	NA
...	...	...	...
http://dbpedia.org/property/dateOfBirth	0	1,20242	NA
http://dbpedia.org/property/profession	0	1,20242	NA

Tabela 6.2: Resultados obtidos relativamente à Entropia, Ganho de Informação e Razão do Ganho de Informação das propriedades utilizadas para a clarificação da multiplicidade de soluções da questão "Who is Barabara Jordan?" - primeira iteração da aplicação do Algoritmo 1.

`//dbpedia.org/resource/Barbara_Jordan_(poet)']` e cada propriedade encontrada é acrescentado ao contexto da questão na forma de facto Prolog

`prop_solution(Referent_Value, Property, Value)`,

que indica o correspondente triplo RDF. No total são acrescentados ao contexto da questão 193 factos que correspondem às propriedades associadas aos referentes da questão.

- O segundo passo é avaliar qual a melhor propriedade para diferenciar os referentes. Este passo consiste em calcular a entropia, o ganho de informação e a taxa de ganho de informação para cada propriedade distinta. Os 193 factos sobre as propriedades e os referentes, que foram colecionados no passo anterior, traduzem apenas informação sobre 58 propriedades distintas. Destas 58 propriedades existem 51 que tem entropia igual a zero. A Tabela 6.2 apresenta os resultados obtidos relativamente à Entropia, Ganho de Informação e Razão do Ganho de Informação das propriedades utilizadas para a clarificação da multiplicidade de soluções da questão "Who is Barabara Jordan?". Os predicados Prolog invocados para efetuar o cálculo da entropia dos dados, a entropia de cada propriedade, o ganho de informação de cada propriedade e a taxa de ganho de informação de cada propriedade que tem entropia diferente de zero são, respetivamente: `data_entropy/3`, `property_entropy/3`, `gain_property/4`, `ratio_gain_property/3`.

- O terceiro passo é escolher a melhor propriedade, i.e., a propriedade que tem maior valor da razão de ganho de informação e cujo valor é expresso em texto. Pela Tabela 6.2 a propriedade a escolher é `http://purl.org/dc/terms/subject`.

- O quarto passo é gerar o conjunto de valores A que correspondem à propriedade `http://purl.org/dc/terms/subject` e que estão no contexto da questão. Ou seja, o conjunto A tem ao todo 34 alternativas e é formado, entre outros, pelos seguintes elementos:

1936 births
1996 deaths
African American members of the United States House of Representatives
African American politicians
African American United States presidential candidates
African American women in politics
Lesbian politicians
1957 births
American female tennis players
Tennis people from Wisconsin
1949 births
American poets
University of Rochester faculty
Living people

Cada elemento do conjunto A forma uma alternativa no diálogo a apresentar ao utilizador.

- O próximo passo é interpretar a escolha do utilizador: Caso o utilizador não saiba escolher uma opção, indicando a opção `?`, o Controlador do Discurso escolherá outra propriedade e apresentará um novo conjunto de alternativas ao utilizador; Caso o utilizador escolha uma das alternativas que leve o sistema a restringir o conjunto de soluções a uma solução este processo é terminado e é fornecida a resposta ao utilizador; Caso o utilizador escolha uma das alternativas (por exemplo, "Living people") que leve o sistema a restringir o conjunto de soluções a duas soluções, o processo continua e volta ao passo 1; Caso o utilizador indicar a opção `!`, o sistema termina este processo e fornecerá todas as respostas; Caso o utilizador indique a expressão `quit`, o sistema termina todo o processo.

▲

Embora o gestor do diálogo ainda esteja em fase de implementação os seguintes predicados Prolog fazem parte dos predicados desenvolvidos nesta fase:

`get_properties(+Referent_list_values)`

Coleciona as propriedades da ontologia que estão relacionadas com os valores dos referentes contidos na lista `Referent_list_values` e adicona cada uma delas ao contexto da questão.

`data_entropy(+T, +Referent_list_values, -Value)`

`Value` é o valor da entropia do conjunto de dados T, relativamente às classes dos dados pertencentes à lista `Referent_list_values`.

`property_entropy(+P, +Referent_list_values, -EntropyValue)`

`EntropyValue` é o valor da entropia da propriedade `P`, para as classes `Referent_list_values`.

`gain_property(S, +Prop, +Referent_list_values, -Gain)`

`Gain` é o ganho de informação da propriedade `P` no conjunto de dados `S` e para as classes da lista `Referent_list_values`.

`ratio_gain_property(+Property_entropy, +Gain_property, -Ratio)`

`Ratio` e a razão do ganho de informação, com `Property_entropy` o valor da entropia da propriedade e `Gain_property` o ganho de informação da mesma propriedade.

6.8 Processamento da Resposta

O Processamento da Resposta, apresentado na Secção 5.3, consiste na determinação da representação final da resposta que será retornada ao utilizador, que é interpretada na base de conhecimento com os factos extraídos. Nesta fase, o Controlador do Discurso analisa a representação do discurso associada à questão (nomeadamente, intenções e crenças do sistema e do utilizador, estrutura e contexto do discurso), conjuntamente com as soluções obtidas e fornece ao utilizador uma resposta adequada, tendo em conta o tipo da questão e a resposta esperada.

Neste momento a resposta consiste apenas em mostrar a solução da questão, sem qualquer cuidado na sua representação. Pelo que, este módulo ainda está numa fase inicial do seu desenvolvimento. O objetivo principal é fornecer uma resposta objetiva, clara, informativa e o mais próximo da língua natural, contendo informação adicional que possa esclarecer possíveis dúvidas ou redirecionar o utilizador na sua pesquisa.

6.9 Interface do Utilizador

O interface do utilizador é bastante elementar, com uma apresentação baseado em linhas de comando. A execução da aplicação é feita através da invocação na linha de comando do SWI-Prolog. Os testes efetuados foram executados num computador *Multi-threaded*, 32 bits e no SWI-Prolog versão 5.10.2..

Como alguns módulos da aplicação não estão implementados, nomeadamente a transformação da questão em língua natural na sua correspondente representação semântica, ou se encontram incompletos, nomeadamente o módulo que processa a questão e gere a sua representação a apresentar ao utilizador, o sistema cooperativo de Pergunta-Resposta para a Web Semântica apresentado ainda não se encontra numa versão que seja possível disponibilizar para utilização dos interessados.

Futuramente, e tendo em conta como principal objetivo as plataformas móveis, pretendemos construir um interface agradável e de fácil utilização, bem como fornecer a sua disponibilização online. Para tal, teremos de finalizar a implementação do sistema com características de um sistema fixo capaz de dialogar em modo de texto, em Inglês. Posteriormente pretendemos estender à língua natural portuguesa. Por último, será definida a arquitetura para plataformas móveis (como por exemplo as que utilizam o sistema *Android* ou o *iOS*) que terá em linha de conta as especificidades destas plataformas, nomeadamente em termos de capacidade de computação e de consumo de energia.

6.10 Conclusão

Neste capítulo, começámos por apresentar a base de conhecimento que considerámos neste trabalho. Depois, apresentamos algumas técnicas e respetivos predicados Prolog desenvolvidos para implementar os diferentes módulos que constituem o sistema cooperativo de Pergunta-Resposta para a Web semântica proposto. Ainda há muito a fazer na conclusão e melhoria da implementação do referido sistema, mas o trabalho feito até ao momento permite-nos testar e avaliar o desempenho do resultado final a que nos propusemos. No próximo capítulo apresentaremos os resultados de uma avaliação concreta da aplicação prática dos procedimentos propostos ao longo dos vários capítulos.

Capítulo 7

Avaliação da Aplicação Prática do Sistema Proposto

A implementação do sistema cooperativo de Pergunta-Resposta para a Web Semântica, apresentado ao longo dos capítulos anteriores, fez parte dos trabalhos inerentes a esta tese. Esta implementação, embora não completa, permitiu a aplicação prática dos procedimentos propostos, cujos resultados podem ser objeto de uma avaliação concreta. O presente capítulo está organizado da seguinte forma. Na Secção 7.1 começamos por apresentar a importância da avaliação de desempenho dos sistemas de Pergunta-Resposta. Na Secção 7.2 apresentamos os sistemas de Pergunta-Resposta que utilizamos para comparação. Posteriormente, na Secção 7.3 apresentamos o conjunto de questões utilizados na avaliação efetuada. Na Secção 7.4, expomos os resultados obtidos na comparação do desempenho dos vários sistemas. Na Secção 7.5, apresentamos os resultados de desempenho obtidos pela nossa proposta. E para finalizar, na Secção 7.6 apresentamos as conclusões.

7.1 Introdução

Os sistemas de Pergunta-Resposta são aplicações bastante complexas. A sua definição e a posterior implementação é normalmente estruturada em vários módulos. A modulação permite que a implementação seja feita de forma separada, incremental e que possa ser melhorada a todo o instante. Para os sistemas de Pergunta-Resposta poderem ser validados e posteriormente serem utilizados necessitam de análises de avaliação do seu desempenho, que suportem opiniões justas sobre eles.

Como vimos nos capítulos anteriores, embora nem todos os módulos do sistema cooperativo de Pergunta-Resposta para a Web Semântica proposto tenham sido implementados, foi possível obter resultados que permitissem a sua avaliação. Para validar o desempenho do sistema proposto, é feita uma comparação com dois sistemas com características semelhantes.

7.2 Sistemas de Pergunta-Resposta Utilizados para Comparação

Na Secção 3.2, apresentámos uma visão geral sobre os sistemas de Pergunta-Resposta para a Web Semântica, bem como um conjunto de sistemas que são semelhantes ao sistema proposto neste trabalho. Estabelecemos as características principais de cada um, as diferenças, como funcionam e, em alguns casos, apresentamos as avaliações feitas pelos autores.

A nossa escolha relativamente a sistemas semelhantes e que seja possível comparar os seus desempenhos com o desempenho do sistema cooperativo de Pergunta-Resposta para a Web Semântica proposto, recaiu apenas em dois sistemas: START e PowerAqua. Esta escolha foi motivado essencialmente por estes sistemas disponibilizarem interfaces web para sua utilização, sem requererem qualquer tipo de especificação, quer de instalação, quer de requisitos, apenas o acesso à Internet.

7.2.1 START - um Sistema de Pergunta-Resposta Baseado na Web

START[1] (*SynTactic Analysis using Reversible Transformations*) [59] é um sistema de Pergunta-Resposta em língua natural que fornece aos utilizadores o acesso a informação multimédia através da utilização de anotações da língua natural. É considerado o primeiro sistema de Pergunta-Resposta baseado na web, e tem estado disponível online e continuamente operacional desde 1993. Foi desenvolvido por Boris Katz e seus associados integrados no grupo InfoLab, pertencente ao departamento de Ciências da Computação e ao Laboratório de Inteligência Artificial da *Massachusetts Institute of Technology* (MIT).

Ao contrário dos sistemas de recuperação de informação (ferramentas de pesquisa), o sistema START tem como principal objetivo fornecer aos seus utilizadores apenas a informação certa, em vez de meramente fornecer uma lista de informação e ligações que poderão conter a resposta pretendida. Atualmente, o sistema é capaz de responder a milhões de questões em Inglês sobre lugares (cidades, países, lagos, coordenadas, tempo, mapas, demográficos, sistemas políticos e económicos), filmes (títulos, atores, diretores), pessoas (datas de nascimento, biografias), definições em dicionários, etc.

O sistema START é uma aplicação desenhada para responder a questões expressas em

[1] `http://start.csail.mit.edu/`

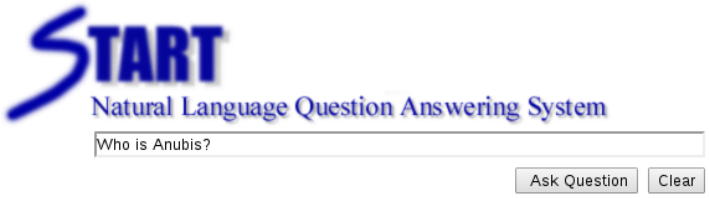


Figura 7.1: Exemplo da questão "Who is Anubis?" a ser colocada ao sistema START, utilizando o interface web disponibilizado online.

língua natural. O sistema interpreta as questões, faz a correspondência entre as consultas geradas através da árvore sintática obtida e a base de conhecimento, apresentando ao utilizador como resposta os segmentos da informação apropriados. Desta forma, oferece aos utilizadores inexperientes acesso rápido ao conhecimento, que em muitos casos um especialista levaria algum tempo a encontrar. As Figuras 7.1 e 7.2 mostram, respetivamente, um exemplo de uma questão ("Who is Anubis?") a ser colocada ao sistema START, utilizando o interface web disponibilizado online, e a respetiva resposta devolvida pelo sistema START, no formato de página web.

A técnica chave, denominada anotações da língua natural, ajuda o sistema START a fazer a ligação de pessoas que procuram determinada informação aos recursos da informação. Esta técnica aplica frases expressas em língua natural como descrições dos conteúdos, que estão associados com os segmentos de informação em várias granularidades. Um segmento de informação é devolvido sempre que a sua anotação corresponde à questão introduzida. Este método permite ao sistema START lidar com toda a variedade de informação: media, texto, diagramas, imagens, vídeos e áudio, conjunto de dados, páginas web, etc.

A componente do sistema START responsável pelo processamento do Inglês é constituída por dois módulos que partilham a mesma gramática: interpretação e geração. O módulo da interpretação analisa o texto escrito em Inglês e produz a base de conhecimento que codifica a informação encontrada no texto. Dado um segmento adequado da base de conhecimento, o módulo de geração produz uma frase também expressa em Inglês. Estes módulos utilizados em conjunto com as técnicas de anotação da língua natural elevam o poder do processamento de língua natural, ao nível das frases, para o uso no serviço de acesso à informação multimédia.

Mais concretamente, o sistema START reformula as questões colocadas pelo utilizador em consultas Omnibase, estabelecendo a ligação entre a língua natural e a estrutura das bases de dados. Obmnibase [58] é uma base de dados que fornece acesso aos recursos web. A base de dados Omnibase impõe o modelo triplo objeto-atributo-valor num conjunto restrito de recursos de dados web. Assim, o sistema START utiliza anotações da língua natural (que são interpretações de afirmações e de frases) que descrevem o conteúdo de vários segmentos de informação. A consulta do utilizador é comparada às anotações armazenadas na base do conhecimento e quando uma correspondência é encontrada, o segmento correspondente à anotação é devolvido ao utilizador, como resposta. Desta forma, as respostas, próximas

START's reply

===> Who is Anubis?

Anubis

- Watchlist
- Uploads
- Settings
- Log in

Anubis (/əˈnuːbəs/ or /əˈnjuːbəs/;[2] Ancient Greek: Ἄνουβις) is the Greek name[3] for a (presumably) jackal-headed god associated with mummification and the afterlife in ancient Egyptian religion. He is the son of Nephthys and Set according to the Egyptian mythology. According to the Akkadian transcription in the Amarna letters, Anubis' name was vocalized in Egyptian as *Anapa*.[4] The oldest known mention of Anubis is in the Old Kingdom pyramid texts, where he is associated with the burial of the pharaoh.[5] At this time, Anubis was the most important god of the dead but he was replaced during the Middle Kingdom by Osiris.[6]

I know about four more terms called "Anubis": Anubis (Stargate), Anubis (Marvel Comics), Anubis (cipher), and 1912 Anubis

Source: Wikipedia

- Go back to the START dialog window.

Figura 7.2: Resposta devolvida pelo sistema START à questão "Who is Anubis?".

da língua natural, tornam-se mais cooperativas e de fácil leitura e compreensão para o utilizador.

7.2.2 PowerAqua - um sistema de Pergunta-Resposta Baseado em Múltiplas Ontologias para a Web Semântica

PowerAqua [70, 71, 69] é um sistema de Pergunta-Resposta baseado em múltiplas ontologias, que tendo por base uma questão expressa em língua natural, é capaz de retornar respostas construídas de recursos relevantes distribuídos na Web Semântica. Em contraste com outros interfaces de língua natural, o sistema PowerAqua não é restrito a uma simples ontologia e, por conseguinte, fornece a primeira tentativa abrangente de suportar domínios abertos como base de conhecimento dos sistemas de Pergunta-Resposta para a Web Semântica.

O sistema PowerAqua evoluiu do anterior sistema denominado AquaLog [72, 74], um sistema de Pergunta-Resposta baseado em ontologias para intranets e limitado ao uso de apenas uma ontologia. O desempenho do AquaLog foi baseado nas medidas de precisão, de recolha e dos tipos de falha. Em média 63,5% de respostas sucessivas são recuperadas a partir da ontologia com ambiente num domínio fechado.

O sistema PowerAqua estende a capacidade fornecida pelo sistema AquaLog, ao suportar

consultas em domínios abertos na Web Semântica. O sistema PowerAqua foi desenhado com o objetivo de tirar proveito do vasto conjunto de dados semânticos heterogéneos oferecidos pela Web Semântica, de forma a interpretar uma consulta, sem fazer previamente qualquer presunção sobre as ontologias relevantes para a consulta efetuada.

O sistema PowerAqua disponibiliza ao utilizador a possibilidade de escolher a ontologia e colocar questões em língua natural, que estejam relacionadas com o domínio coberto pela ontologia escolhida. A arquitetura do sistema e os métodos de raciocínio utilizados são independentes do domínio, dependendo da semântica da ontologia, e faz uso de recursos lexicais genéricos, tais como a WordNet. O sistema é capaz de aprender a terminologia do utilizador de forma a melhorar a sua experiência no tempo, i.e., o sistema tem mecanismos que permitem guardar as questões e respetivas respostas e posteriormente poderem ser utilizadas de uma forma mais rápida e eficiente. O seu mecanismo de aprendizagem utiliza o raciocínio sobre a ontologia para aprender padrões mais genéricos, que poderão depois ser reutilizados para perguntas com contextos semelhantes.

Neste sistema são utilizados dois modelos importantes: a componente linguística, que é aplicada na conversão das questões expressas em língua natural na correspondente forma linguística; e o serviço de relação de similaridade (*Relation Similarity Service*, RSS) que transforma a forma linguística num formato de triplos na ontologia. Ou seja, os termos linguísticos são substituídos pelos termos equivalentes na ontologia. O modelo de dados é baseado em triplos, do tipo `Sujeito Predicado Objeto`.

Os resultados obtidos pelo sistema PowerAqua apenas conduzem a respostas parciais, i.e., para cada formato triplo da ontologia é fornecido pelo sistema as respostas correspondentes. Pelo que é necessário serem combinados de forma a obter uma resposta completa. As respostas são apresentadas ao utilizador agrupadas e classificadas de acordo com as várias interpretações, produzidas pelas diferentes ontologias.

As Figuras 7.3 e 7.4 mostram, respetivamente, um exemplo de uma questão ("Who is Anubis?") a ser colocada ao sistema PowerAqua-DBpedia, utilizando o interface web disponibilizado online, e a respetiva resposta devolvida pelo sistema PowerAqua-Dbpedia, no formato de página web.

7.3 Conjunto de Questões Utilizado na Avaliação

A avaliação do Controlador do Discurso apresentado, bem como do sistema de Pergunta-Resposta no seu todo [84], envolve o uso de uma base de conhecimento constituída apenas pela ontologia OWL2 da DBpedia [19], que cobre cerca de 359 classes, formando uma hierarquia de subsunção, e descritas por 1.775 propriedades diferentes. A ontologia da DBpedia contém cerca de 2.350.000 instâncias. Para tornar a base de conhecimento mais completa, é usado também a ferramenta SPARQL[2] *endpoints* para questionar a base de

[2]http://www.w3.org/TR/rdf-sparql-query/

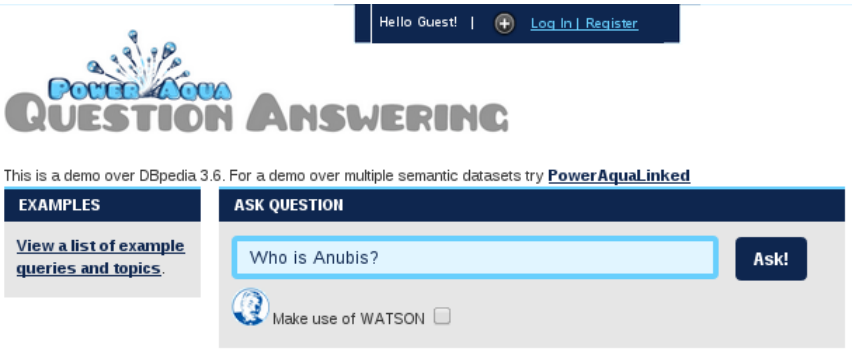


Figura 7.3: Exemplo da questão "Who is Anubis?" a ser colocada ao sistema PowerAqua-DBpedia, utilizando o interface web disponibilizado online.

dados DBpedia RDF[3] e o serviço de pesquisa online da DBpedia, *Lookup Service* [4], que permite encontrar URIs [5] relacionadas com uma determinada palavra-chave. Relacionada significa que tanto o rótulo de um recurso coincide, como um texto que é utilizado frequentemente na Wikipédia para se referir a um recurso específico coincide (por exemplo, o recurso `http://dbpedia.org/resource/United_States` pode ser procurado através da string "USA"). Os resultados são ordenados utilizando o número de ligações apontadas por outras páginas da Wikipédia e apresentados numa página de resultados.

O teste de avaliação foi feito utilizando um conjunto de 84 questões apresentado no TREC 9 (*The Ninth Text REtrieval Conference* [108]). O conjunto em análise é constituído apenas por questões do tipo "wh" diretas, fazendo parte as seguinte questões:

```
222. Who is Anubis?
232. Who invented television?
347. Who was Monet?
390. Where was John Adams born?
459. When was John D. Rockefeller born?
534. Where is Windsor Castle?
773. What city is Logan Airport in?
759. What is the collective noun for geese?
```

7.4 Resultados Obtidos na Comparação com os Sistemas de Pergunta-Resposta START e PowerAqua

O conjunto de questões foi utilizado para avaliar o desempenho na obtenção de respostas por parte do nosso sistema de Pergunta-Resposta e também para comparar com dois outros

[3] http://www.w3.org/TR/REC-rdf-syntax/
[4] `http://wiki.dbpedia.org/lookup/`
[5] `http://tools.ietf.org/html/rfc3986/`

Figura 7.4: Resposta devolvida pelo sistema PowerAqua-DBpedia à questão "Who is Anubis?".

sistema: START e PowerAqua (apenas com a base de conhecimento DBpedia[6]).

Os testes foram executados individualmente e todas as respostas obtidas pelo sistema foram verificadas manualmente, de forma a garantirmos a sua veracidade.

A Tabela 7.1 apresenta o desempenho do nosso sistema e dos outros dois sistemas, relativamente ao número de respostas não obtidas, respostas corretas, respostas incorretas e exatidão das respostas devolvidas. Exatidão refere-se à proporção do total de respostas corretas relativamente ao número total de questões.

Os valores apresentados na Tabela 7.1 demonstram que podemos considerar a nossa proposta com um desempenho bastante próximo dos resultados dos outros sistemas no que concerne à exatidão. Ou seja, 76% das respostas obtidas pelo nosso sistema são corretas contra o melhor resultado 80% obtido pelo sistema START. Estes valores levam-nos a clamar o evidente sucesso da nossa proposta e que nos encorajam a prosseguir na melhoria do seu desempenho.

[6] `http://poweraqualinked.open.ac.uk:8080/poweraquadbpedia2`

	Nenhuma resposta	**Resposta correta**	**Resposta Errada**	**Exatidão**
Nossa proposta	15	64	5	76%
START	8	67	9	80%
PowerAqua-DBpedia	3	66	15	79%

Tabela 7.1: Informação dos resultados obtidos no teste de desempenho da nossa proposta e dos sistemas START e PowerAqua-DBpedia.

O número de resposta não encontradas pelo nosso sistema é bastante alto, 15 (18% das questões), relativamente ao desempenho dos outros sistema, melhor caso obtido pelo sistema PowerAqua corresponde a 3 (4% das questões). Este mau desempenho atribui-se a algumas falhas encontradas:

- na Descoberta da Ontologia - onde a pesquisa pelos termos na ontologia equivalentes aos termos linguísticos da questão não contempla todos os casos, condicionada pela interpretação sintática da questão;
- na Avaliação Semântica - onde a interpretação na ontologia da DRS da questão pode excluir alguns casos que levarão à resposta correta, condicionada pela representação semântica da questão;
- na base de conhecimento - não uniformização das propriedades contempladas por cada entidade da base de conhecimento. Ao contrário das bases de dados, o conjunto dos atributos de cada recurso da ontologia pode não ser igual e a informação associada a cada entidade pode estar incompleta.

A solução passa por melhorar as regras que permitem mapear os termos linguísticos nas entidades da ontologia; aumentar a ontologia, por adição de outras ontologias que possam definir mais conceitos e alargar o leque de informação de cada entidade; aumentar a base de conhecimento, por adição de novas regras de inferência, novas técnicas de pesquisa e validação, que permitam aumentar o leque de informação sobre uma determinada questão.

Adicionalmente, pretendemos no futuro analisar e testar a aplicação de técnicas de relaxação de restrições aquando da procura e validação dos recursos na base de conhecimento. Caso seja possível melhorar o desempenho do sistema, no que diz respeito à diminuição dos casos de resposta vazia ou resposta errada, o sistema será enriquecido com a implementação das técnicas referidas.

Outro problema encontrado foi o tempo de execução, que não está a ser atualmente avaliado. O sistema demora vários minutos para responder às questões testadas e se aumentarmos a complexidade das questões, o sistema poderá demorar várias horas para obter uma resposta. Muita da pesquisa é feita online e todas as hipóteses são testadas (tanto para as entidades da ontologia como na execução das regras de inferência na avaliação semântica), pelo que o desempenho do nosso sistema não é particularmente eficiente na obtenção de

	Total	Relevância
Não obteve resposta	15	18%
Respostas corretas	64	76%
Simples	20	24%
Múltiplas	44	52%
Respostas não corretas	5	6%
Simples	1	1%
Múltiplas	4	5%
Média das soluções múltiplas	3,6166667	

Tabela 7.2: Informação dos Resultados obtidos na Avaliação do Desempenho do sistema de Pergunta-Resposta para a Web Semântica proposto.

respostas em tempo real (um importante aspeto nos sistemas de Pergunta-Resposta para a Web Semântica).

7.5 Resultados Obtidos no Desempenho do Sistema de Pergunta-Resposta para a Web Semântica proposto

Numa primeira análise dos resultados, apresentados na Tabela 7.2, começámos por observar que o sistema não obteve resposta a 15 questões (18% das questões). Ou seja, o sistema não encontrou, na base de conhecimento, os recursos que identificam os termos da questão ou as entidades que são solução para a questão. Por exemplo, na questão 759, o sistema não encontrou recursos que relacionassem os termos "collective noun" e "geese". A abertura de um diálogo com o utilizador permitirá que este possa reescrever a questão colocada, esclarecer os termos ou mesmo colocar uma nova questão. Desta forma, caso a base de conhecimento contenha a resposta à questão colocada, poderemos aumentar claramente o sucesso do sistema. Analisando o restante corpus, reduzido a 69 questões, obtivemos 64 respostas corretas (76% das questões), que foram verificadas uma a uma manualmente. Dentro destas, 44 questões eram de resposta múltipla (52% das questões) que, com o esclarecimento do utilizador, o sistema devolveu a resposta esperada. Constatámos que, para cada questão de resposta múltipla, o sistema obteve em média 3 a 4 soluções. Manifestamente um conjunto reduzido de alternativas, evidenciando o potencial do sistema na procura da resposta. Às restantes 5 questões, o sistema não obteve a resposta correta (6% das questões). Estas falhas identificam-se com alguns fatores que originam interpretações incorretas, nomeadamente: representação semântica da questão; questões incompletas ou mal formuladas; dimensão da base do conhecimento; ou informação incompleta e não uniforme dos recursos da ontologia.

Por exemplo, na questão `222. Who is Anubis?`, a entidade que representa "Anubis" está semanticamente relacionada com o facto de ser uma pessoa, o que condiciona a resposta a obter. Ou seja, o sistema terá de encontrar as entidades na base de conhecimento que sejam pessoas e cujo nome seja "Anubis". No entanto, o recurso da base de conhecimento, que origina a resposta correta `http://dbpedia.org/resource/Anubis`, não tem qualquer relação com a condição de ser pessoa. Pelo que o sistema é influenciado no percurso que levará à obtenção de uma resposta incorreta, motivado tanto pela representação semântica da questão, como pela informação incompleta dos recursos da base de conhecimento. Como dissemos anteriormente, este problema pode ser resolvido se aplicarmos técnicas de relaxação das restrições na procura dos recurso na ontologia e na validação das restrições na base de conhecimento. Ou seja, se no exemplo relaxarmos a condição de que "Anubis" é uma pessoa, o leque de soluções irá aumentar e incluir a entidade que é solução da questão.

7.6 Conclusão

A experimentação num conjunto pequeno de perguntas simples e diretas do tipo "wh" mostrou que o nosso sistema produz resultados promissores. À parte o tempo de execução, também testámos algumas questões mais complexas e o sistema portou-se como pretendíamos: devolveu as respostas a essas questões. Acreditamos que adicionar uma ferramenta, tal como o Controlador do Discurso, a um sistema de Pergunta-Resposta permite melhorar substancialmente o desempenho do sistema. Recorrer a um diálogo controlado para clarificar ambiguidades, ajuda o sistema a interpretar melhor as pretensões do utilizador. Estes diálogos permitem aumentar os resultados obtidos pelo sistema e ajudam-no a gerar uma resposta adequada, mais objetiva, clara e com a informação pretendida pelo utilizador. Uma vez que um dos nossos objetivos é gerar as respostas expressas em língua natural e não fornecer uma lista de resultados, pensamos que com os testes feitos mostramos o sucesso da nossa proposta. Pelo que, acreditamos que a nossa proposta se aproxima rapidamente da proposta que ajuda a preencher a lacuna entre a Web Semântica baseada em lógica e os utilizadores do mundo real.

A avaliação apresentada neste capítulo é ainda muito preliminar, resumindo-se apenas a um primeiro conjunto de testes, cujos resultados são bastante promissores, permitindo verificar a eficácia do sistema de Pergunta-Resposta para a Web Semântica proposto e identificar as deficiências que permitirão melhorar a seu desempenho. Futuramente pretendemos apresentar uma avaliação mais completa, estender o conjunto de questões aos outros tipos e incluir a avaliação do tempo de execução, fornecendo valores comparativos utilizando resultados extraídos de outros sistemas de Pergunta-Resposta semelhantes. No entanto, os resultados obtidos encorajam-nos a prosseguir.

Capítulo 8

Conclusões e Trabalho Futuro

O aumento exponencial de informação não estruturada e heterogénea disponível na web, a dificuldade de identificar, interpretar e contextualizar esta informação; o vasto leque de desafios, tais como: contexto, classes e processamento das questões, extração e formulação das respostas em tempo real; a crescente necessidade de motores de busca inteligentes capazes de satisfazer as exigências dos mais variados tipos de utilizadores da web; a necessidade de interação e cooperação entre os utilizadores e os sistemas; a necessidade de apresentar, por parte do sistema, respostas precisas, informativas e expressas o mais próximo da língua natural; são fatores que tornam o problema de desenvolvimento de sistemas de Pergunta-Resposta para a Web Semântica bastante complexo.

Ao longo dos tempos, os sistemas de pesquisa na web têm vindo a tornar-se indispensáveis nas mais variadas utilizações da web. No entanto, escasseiam as ferramentas capazes de "dialogar" com o comum utilizador e que respondam de forma eficaz às suas necessidades imediatas. O presente trabalho foi desenvolvido com o intuito de colmatar alguns dos problemas associados à pesquisa na web.

Na presente tese começámos por definir os conceitos "ontologias" e "Web Semântica", evidenciando a forma como estão relacionados no contexto da consulta na web. Em adição, apresentamos como estes dois conceitos suportam o desenvolvimento do nosso trabalho através de linguagens de representação. Introduzimos também outros dois conceitos "Estruturas de Representação do Conhecimento" e "Processamento de Língua Natural" inerentes ao nosso trabalho.

Definimos os sistemas de Pergunta-Resposta, focando os trabalhos desenvolvidos que estão relacionados com sistemas cooperativos e têm como domínio de conhecimento a Web Semântica. Apresentamos a arquitetura típica de um sistema de Pergunta-Resposta, in-

troduzimos algumas características sobre metodologias utilizadas mais frequentemente nos sistemas de Pergunta-Resposta, enumeramos um conjunto de desafios inerentes ao desenvolvimento de sistemas de Pergunta-Resposta, apresentamos alguns tópicos de pesquisa atuais.

Apresentamos a nossa proposta de sistema cooperativo de Pergunta-Resposta para a Web Semântica que recebe questões expressas em língua natural e é capaz de devolver uma resposta cooperativa, também expressa em língua natural, obtida da base de conhecimento. Por último, também abordamos e avaliamos os resultados da aplicação prática a que submetemos o sistema apresentado.

Um dos pontos inovadores do trabalho desenvolvido é o sistema estar enriquecido com um Controlador do Discurso que, através da análise da questão e do tipo de resposta esperada, da representação semântica das questões, a estrutura do discurso que contempla as intenções do utilizador e o contexto das questões, permite fornecer respostas precisas às questões colocadas pelo utilizador em língua natural.

O nosso sistema é um sistema cooperativo de Pergunta-Resposta simples e *friendly*. Nesta fase, não pretendemos que a nossa proposta seja um sistema inteligente completo por interpretar e entender as questões colocadas pelos utilizadores. Ele tira partido de um conjunto reduzido de técnicas de Processamento de Língua Natural (tais como, estruturas gramaticais, Estruturas de Representação do Conhecimento, WordNet), regras de inferência e inicia um diálogo controlado com o utilizador quando não consegue continuar com o processo de alcançar a resposta.

Outro ponto inovador do trabalho desenvolvido é a capacidade cooperativa do sistema: a capacidade de interação com o utilizador de forma a obter informação necessária para esclarecer e conduzir o sistema no caminho para a resposta correta; fornecer uma resposta direta, precisa, clara, objetiva e contendo informação adicional que possa elucidar melhor o utilizador.

O trabalho desenvolvido, ao longo dos últimos 3 anos, foi sendo reconhecido pela comunidade científica. Numa primeira fase, a nossa proposta de sistema cooperativo de Pergunta-Resposta para a Web Semântica apresentada nos artigos [85, 86], foram aceites e apresentados, respetivamente, na Conferência Internacional *KMIS 2011 - International Conference on Knowledge Management and Information Sharing* e nas 2.ªs Jornadas de Informática da Universidade de Évora - JIUE'2011. No seguimento destes artigos foram elaborados 2 artigos [89, 87] que foram aceites para publicação, respetivamente, nas revistas científicas internacionais *Advances in Artificial Intelligence* e *International Journal of Computational Linguistics and Applications*. Destes artigos, o segundo foi aceite previamente na Conferência Internacional *Computational Linguistics and Intelligent Text Processing - 13th International Conference, CICLing 2012* [88], sendo posteriormente selecionado para publicação na referida revista. Os resultados da avaliação feita ao sistema foram alvo do artigo [84] que foi aceite para apresentação no 4.º Simpósio de Informática - INForum 2012, tendo sido publicado nas respetivas Atas. Mais recentemente, a revisão

apresentada sobre os sistemas de Pergunta-Resposta constitui o principal tema do artigo [90], que foi aceite e apresentado nas 3.ªs Jornadas de Informática da Universidade de Évora - JIUE'2013 e posteriormente publicado nas respetivas Atas.

Como trabalho futuro, pretendemos:

- Finalizar a implementação do sistema com características de um sistema fixo com capacidade de dialogar em modo de texto, na língua natural Inglesa;
- Construir um interface agradável e de fácil utilização, bem como fornecer a sua disponibilização online;
- Melhorar as técnicas de pesquisa na ontologia e as regras de inferência com o objetivo de melhorar os resultados de exatidão e os tempos de execução;
- Aumentar o conjunto de testes, que abranjam os restantes tipos de questões (incluindo questões mais complexas) e respostas esperadas e definir uma avaliação quantitativa, qualitativa e comparativa mais completa do desempenho tanto do sistema como do Controlador do Diálogo, utilizando resultados extraídos de outros sistemas de pergunta-resposta;
- Estender o sistema à língua Portuguesa. Para este efeito, será necessário enriquecer o domínio do conhecimento com conceitos que possam ser deduzidos a partir do domínio inicial;
- Ampliar o domínio do conhecimento com outras ontologias que permitam suportar o conceito de domínio aberto e aproveitar a grande quantidade de informação semântica heterogénea fornecida pela Web Semântica;
- Definir a arquitetura para plataformas móveis (como por exemplo as que utilizam o sistema *Android* ou o *iOS*) que terá em linha de conta as especificidades destas plataformas, nomeadamente em termos de capacidade de computação e de consumo de energia.

Bibliografia

[1] Agichtein, E., Castillo, C., Donato, D., Gionis, A., and Mishne, G. Finding high-quality content in social media. In *Proceedings of the 2008 International Conference on Web Search and Data Mining* (New York, NY, USA, 2008), WSDM '08, ACM, pp. 183–194.

[2] Allam, A., and Haggag, M. The Question Answering Systems: A Survey. *International Journal of Research and Reviews in Information Sciences 2*, 3 (2012), 211–221.

[3] Allen, J. F. Maintaining knowledge about temporal intervals. *Commun. ACM 26* (November 1983), 832–843.

[4] Andrenucci, A., and Sneiders, E. Automated question answering: Review of the main approaches. In *ICITA (1)* (2005), IEEE Computer Society, pp. 514–519.

[5] Auer, S., Bizer, C., Kobilarov, G., and Lehmann, J. Dbpedia: A nucleus for a web of open data. *The Semantic Web 4825*, Springer (2007), 722–735.

[6] Azzam, S., and Humphreys, K. New Directions in Question Answering. *Information Retrieval 9*, 3 (June 2006), 383–386.

[7] Baader, F. Description logics. In *Reasoning Web: Semantic Technologies for Information Systems, 5th International Summer School 2009*, vol. 5689 of *Lecture Notes in Computer Science*. Springer–Verlag, 2009, pp. 1–39.

[8] Baader, F., Küsters, R., and Wolter, F. The description logic handbook. Cambridge University Press, New York, NY, USA, 2003, ch. Extensions to description logics, pp. 219–261.

[9] Baader, F., and Nutt, W. The description logic handbook. Cambridge University Press, New York, NY, USA, 2003, ch. Basic description logics, pp. 43–95.

[10] Baader, F., and Sattler, U. Description logics with aggregates and concrete domains. *Inf. Syst. 28* (December 2003), 979–1004.

[11] Baeza-Yates, R. A., and Ribeiro-Neto, B. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[12] Barwise, J., and Cooper, R. Generalized quantifiers and natural language. *Linguistics and philosophy 4*, 2 (1981), 159–219.

[13] Benamara, F. A semantic representation formalism for cooperative question answering systems. In *Proceeding of Knowledge Base Computer Systems (KBCS)* (Mumbai, Narosa, Delhi, Inde, December 2002), pp. 221–229.

[14] Benamara, F. Cooperative question answering in restricted domains: the web-coop experiment. In *ACL Worshop on Question Answering in Restricted Domains* (Barcelona, Spain, July 2004), D. M. Aliod and J. L. Vicedo, Eds., Association for Computational Linguistics, pp. 31–38.

[15] Benamara, F. Generating intensional answers in intelligent question answering systems. In *Natural Language Generation*, A. Belz, R. Evans, and P. Piwek, Eds., vol. 3123 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2004, pp. 11–20.

[16] Benamara, F. A semantic representation formalism for cooperative question answering systems. In *Proceeding of Knowledge Base Computer Systems (KBCS)* (2008).

[17] Benamara, F., and Saint-Dizier, P. Lexicalisation strategies in cooperative question-answering systems. In *Proceedings of the 20th international conference on Computational Linguistics* (Stroudsburg, PA, USA, 2004), COLING '04, Association for Computational Linguistics.

[18] Bian, J., Liu, Y., Agichtein, E., and Zha, H. Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of the 17th international conference on World Wide Web* (New York, NY, USA, 2008), WWW '08, ACM, pp. 467–476.

[19] Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. DBpedia-A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web 7*, 3 (2009), 154–165.

[20] Blackburn, P., and Bos, J. *Representation and inference for natural language: A first course in computational semantics*. Center for the Study of Language and Information, 2005.

[21] Borgida, and Brachman. Conceptual Modeling with Description Logics. In *The Description Logic Handbook: Theoryand Implementationand and Applications*,

F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, Eds. Cambridge University Press, 2003, pp. 359–381.

[22] Borgida, A. Description logics in data management. *Knowledge and Data Engineering, IEEE Transactions on 7*, 5 (Oct. 1995), 671 –682.

[23] Borgida, A. On the relative expressiveness of description logics and predicate logics. *Artif. Intell. 82* (April 1996), 353–367.

[24] Burger, J., Cardie, C., Chaudhri, V., Gaizauskas, R., Harabagiu, S., Israel, D., Jacquemin, C., Lin, C. Y., Maiorano, S., Miller, G., and Others. Issues, tasks and program structures to roadmap research in question & answering (Q&A). *Document Understanding Conferences Roadmapping Documents* (2001), 1–35.

[25] Candan, K. S., Liu, H., and Suvarna, R. Resource description framework: metadata and its applications. *SIGKDD Explor. Newsl. 3*, 1 (July 2001), 6–19.

[26] Carpineto, C., and Romano, G. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys 44*, 1 (Jan. 2012), 1:1–1:50.

[27] Chandrasekaran, B., Josephson, J. R., and Benjamins, V. R. What are ontologies, and why do we need them? *IEEE Intelligent Systems 14*, 1 (Jan. 1999), 20–26.

[28] Corella, F., and Lewison, K. A brief overview of cooperative answering. *Journal of Intelligent Information Systems 1*, 2 (Oct. 2009), 123–157.

[29] Damljanovic, D., Agatonovic, M., and Cunningham, H. Natural language interfaces to ontologies: combining syntactic analysis and ontology-based lookup through the user interaction. In *Proceedings of the 7th international conference on The Semantic Web: research and Applications - Volume Part I* (Berlin, Heidelberg, 2010), ESWC'10, Springer-Verlag, pp. 106–120.

[30] Damljanovic, D., Agatonovic, M., and Cunningham, H. Freya: An interactive way of querying linked data using natural language. In *The Semantic Web: ESWC 2011 Workshops*, R. García-Castro, D. Fensel, and G. Antoniou, Eds., vol. 7117 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 125–138.

[31] de Sena, G. J., and Furtado, A. L. Towards a cooperative question-answering model. *Flexible Query Answering Systems 1495* (1998), 354–365.

[32] Donini, F. M. The description logic handbook. Cambridge University Press, New York, NY, USA, 2003, ch. Complexity of reasoning, pp. 96–136.

[33] Fellbaum, C., Ed. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*, illustrated edition ed. The MIT Press, May 1998.

[34] Franz Baader, Ian Horrocks, and Ulrike Sattler. Description Logics. In *Handbook of Knowledge Representation*, V. L. Frank van Harmelen and B. Porter, Eds. Elsevier, 2007, ch. 3.

[35] Gaasterland, T. Cooperative answering through controlled query relaxation. *IEEE Expert: Intelligent Systems and Their Applications 12*, 5 (Sept. 1997), 48–59.

[36] Gaasterland, T., and Godfrey, P. Relaxation as a platform for cooperative answering. *Journal of Intelligent Information 1*, 3 (1992), 293–321.

[37] Gaasterland, T., Godfrey, P., and Minker, J. An overview of cooperative answering. *Journal of Intelligent Information Systems 1*, 2 (1992), 123–157.

[38] Gaasterland, T., Godfrey, P., Minker, J., and Novik, L. A cooperative answering system. In *Logic Programming and Automated Reasoning* (1992), no. X, Springer, pp. 478–480.

[39] Gruber, T. R. A translation approach to portable ontology specifications. *Knowl. Acquis. 5*, 2 (June 1993), 199–220.

[40] Gruber, T. R. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human Computer Studies 43*, 5-6 (Dec. 1995), 907–928.

[41] Guo, Q., and Zhang, M. Question answering based on pervasive agent ontology and Semantic Web. *Knowledge-Based Systems 22*, 6 (Aug. 2009), 443–448.

[42] Haarslev, V., and Möller, R. Description of the racer system and its applications. In *International workshop on Description Logics (DL-2001)* (Stanford, August 2001).

[43] Haarslev, V., and Möller, R. Racer system description. In *Proceedings of the first International Joint Conference on Automated Reasoning (IJCAR'01)* (2001), R. Goré, A. Leitsch, and T. Nipkow, Eds., no. 2083, Lecture Notes in Artificial Intelligence, Springer-Verlag, pp. 701–705.

[44] Hao, T., Liu, W., and Agichtein, E. Towards automatic question answering over social media by learning question equivalence patterns. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media* (Stroudsburg, PA, USA, 2010), WSA '10, Association for Computational Linguistics, pp. 9–10.

[45] Hieber, F., and Riezler, S. Improved answer ranking in social question-answering portals. In *SMUC* (2011), I. Cantador, F. M. Carrero, J. C. Cortizo, P. Rosso, M. Schedl, and J. A. Troyano, Eds., ACM, pp. 19–26.

[46] Hirschman, L., and Gaizauskas, R. Natural language question answering: The view from here. *Natural Language Engineering 7*, 4 (2001), 275–300.

[47] Hodges, W. Classical logic I: first-order logic. *The Blackwell guide to philosophical logic* (2001), 9–32.

[48] Hoffart, J., Suchanek, F. M., Berberich, K., Lewis-Kelham, E., de Melo, G., and Weikum, G. Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World wide web* (New York, NY, USA, 2011), WWW '11, ACM, pp. 229–232.

[49] Horrocks, I. Ontologies and the semantic web. *Communications of the ACM 51*, 12 (Dec. 2008), 58.

[50] Horrocks, I., Patel-Schneider, P. F., and Van Harmelen, F. From SHIQ and RDF to OWL: The making of a web ontology language. *Web semantics: science, services and agents on the World Wide Web 1*, 1 (2003), 7–26.

[51] Horroks, I. Using a expressive description logic: Fact or fiction? In *Proceedings of the Sixth Internacional Conference on Principles of Knowledge Representation and Reasoning (KR98)* (1998), pp. 636–647.

[52] Jousse, F., Tellier, I., Tommasi, M., and Marty, P. Learning to extract answers in question answering: Experimental studies. In *CORIA* (2005), p. 85.

[53] Jurafsky, D., and Martin, J. H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Prentice Hall Series in Artificial Intelligence)*, 1 ed. Prentice Hall, Feb. 2000.

[54] Kamp, H. Discourse representation theory. In *Handbook of Pragmatics*, J. Verschueren, J.-O. Östman, and J. Blommaert, Eds. Benjamins, 1995, pp. 253–257.

[55] Kamp, H., Genabith, J., and Reyle, U. Discourse representation theory. In *Handbook of Philosophical Logic*, D. M. Gabbay and F. Guenthner, Eds., vol. 15 of *Handbook of Philosophical Logic*. Springer Netherlands, 2011, pp. 125–394.

[56] Kamp, H., and Reyle, U. *From Discourse to Logic*, vol. 42 of *Studies in Linguistics and Philosophy*. Kluwer, 1993.

[57] Kamp, H., and Reyle, U. *Semantics: An International Handbook of Natural Language Meaning*. de Gruyter, 2011, ch. Discourse Representation Theory, pp. 872–919.

[58] Katz, B., Felshin, S., Yuret, D., Ibrahim, A., Lin, J. J., Marton, G., McFarland, A. J., and Temelkuran, B. Omnibase: Uniform access to heterogeneous data for question answering. In *Proceedings of the 6th International Conference on Applications of Natural Language to Information Systems-Revised Papers* (London, UK, UK, 2002), NLDB '02, Springer-Verlag, pp. 230–234.

[59] Katz, B., Lin, J. J., and Felshin, S. The start multimedia information system: Current technology and future directions. In *Multimedia Information Systems* (2002), MIS 2002, International Workshop on Multimedia Information Systems, October 10 - November 1, 2002, Tempe, Arizona, USA, Proceedings, Arizona State University, pp. 117–123.

[60] Kaufmann, E., and Bernstein, A. Evaluating the usability of natural language query languages and interfaces to Semantic Web knowledge bases. *Web Semantics: Science, Services and Agents on the World Wide Web 8*, 4 (Nov. 2010), 377–393.

[61] Kaufmann, E., Bernstein, A., and Zumstein, R. Querix: A natural language interface to query ontologies based on clarification dialogs. In *5th International Semantic Web Conference (ISWC 2006)* (November 2006), Springer, pp. 980–981.

[62] Klein, D., and Manning, C. D. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1* (Stroudsburg, PA, USA, 2003), ACL '03, Association for Computational Linguistics, pp. 423–430.

[63] Krifka, M. Questions. In *Semantics: An international handbook of Natural Language Meaning*, P. P. Heusinger, Klaus von, Claudia Maienborn, Ed., vol. 2. Berlin, 2011, ch. 66, pp. 1742–1785.

[64] Kumar, A., and Sebastian, T. M. Sentiment Analysis: A Perspective on its Past, Present and Future. *International Journal of Intelligent Systems and Applications 4*, 10 (Sept. 2012), 1–14.

[65] Lee, T. B., Hendler, J., Lassila, O., and Others. The semantic web. *Scientific American 284*, 5 (May 2001), 34–43.

[66] Leidner, J. L. Handbook of natural language processing. *Computational Linguistics 37*, 2 (2011), 395–397.

[67] Lin, J. The Web as a resource for question answering: Perspectives and challenges. In *Proceedings of the Third International Conference on Language Resources and Evaluation* (2002), no. Lrec, Citeseer, pp. 2120–2127.

[68] Liu, B., and Zhang, L. A survey of opinion mining and sentiment analysis. In *Mining Text Data*, C. C. Aggarwal and C. Zhai, Eds. Springer, 2012, pp. 415–463.

[69] Lopez, V., Fernández, M., Motta, E., Sabou, M., and Uren, V. Question Answering on the Real Semantic Web. In *6th International and 2nd Asian Semantic Web Conference (ISWC 2007+ ASWC 2007)* (2007), pp. 2–4.

[70] Lopez, V., Fernández, M., Motta, E., and Stieler, N. Poweraqua: Supporting users in querying and exploring the semantic web. *Semantic Web 3*, 3 (2012), 249–265.

[71] Lopez, V., Motta, E., and Uren, V. S. Poweraqua: Fishing the semantic web. In *ESWC* (2006), Y. Sure and J. Domingue, Eds., vol. 4011 of *Lecture Notes in Computer Science*, Springer, pp. 393–410.

[72] Lopez, V., Pasin, M., and Motta, E. Aqualog: An ontology-portable question answering system for the semantic web. In *ESWC* (2005), A. Gómez-Pérez and J. Euzenat, Eds., vol. 3532 of *Lecture Notes in Computer Science*, Springer, pp. 546–562.

[73] Lopez, V., Uren, V., Sabou, M., and Motta, E. Is question answering fit for the semantic web?: a survey. *Semantic Web? Interoperability, Usability, Applicability 2*, 2 (September 2011), 125–155.

[74] Lopez, V., Uren, V. S., Motta, E., and Pasin, M. Aqualog: An ontology-driven question answering system for organizational semantic intranets. *Journal of Web Semantics 5*, 2 (2007), 72–105.

[75] Lutz, C. Adding numbers to the $\mathcal{SHIQ}$ description logic - first results. In *Proceedings of the Eighth Internacional Conference on Principles of Knowledge Representation and Reasoning (KR2002)* (2002), Morgan Kaufman, pp. 191–202.

[76] Lutz, C. Description logics with concrete domains-a survey. In *Advances in Modal Logic'02* (2002), pp. 265–296.

[77] Lutz, C. Combining interval-based temporal reasoning with general tboxes. *Artif. Intell. 152* (February 2004), 235–274.

[78] Lutz, C., Sturm, H., Wolter, F., and Zakharyaschev, M. A tableau calculus for temporal description logic: The constant domain case. In *Proceedings of the first International Joint Conference on Automated Reasoning (IJCAR'01)* (2001), R. Goré, A. Leitsch, and T. Nipkow, Eds., no. 2083, Lecture Notes in Artificial Intelligence, Springer-Verlag, pp. 121–136.

[79] Manning, C. D., Raghavan, P., and Schütze, H. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[80] Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. Building a large annotated corpus of english: the penn treebank. *Comput. Linguist. 19*, 2 (June 1993), 313–330.

[81] Maybury, M. New directions in question answering. In *Advances in Open Domain Question Answering*, T. Strzalkowski and S. Harabagiu, Eds., vol. 32 of *Text, Speech and Language Technology*. Springer Netherlands, 2006, pp. 533–558.

[82] Mcguinness, D. L. Question Answering on the Semantic Web. *IEEE Intelligent Systems* (2004), 6–9.

[83] Melo, D. *Ferramentas para determinação e avaliação de soluções em problemas de horários*. Tese de Mestrado em Informática, Departamento de Ciência de Computadores, Faculdade de Ciências, Universidade do Porto, Dezembro de 2006.

[84] Melo, D., Rodrigues, I., and Nogueira, V. Um sistema de pergunta-resposta para ontologias owl. In *Actas do INForum 2012 - 4.º Simpósio de Informática* (Monte da Caparica, Portugal, September, 6-7 2012), A. Lopes and J. O. Pereira, Eds., Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa.

[85] Melo, D., Rodrigues, I. P., and Nogueira, V. B. Cooperative question answering for the semantic web. In *KMIS 2011 - Proceedings of the International Conference on Knowledge Management and Information Sharing* (Paris, France, October, 26-29 2011), J. Filipe and K. Liu, Eds., pp. 258–263.

[86] Melo, D., Rodrigues, I. P., and Nogueira, V. B. Cooperative Question Answering for the Semantic Web. In *Actas das Jornadas de Informática da Universidade de Évora 2011* (Departamento de Informática, Escola de Ciências e Tecnologia da Universidade de Évora, November, 16 2011), L. Rato and T. Gonçalves, Eds., Escola de Ciências e Tecnologia, Universidade de Évora, pp. 1–6.

[87] Melo, D., Rodrigues, I. P., and Nogueira, V. B. Puzzle out the semantic web search. *International Journal of Computational Linguistics and Applications 3*, 1 (June 2012), 91–106.

[88] Melo, D., Rodrigues, I. P., and Nogueira, V. B. Puzzle out the semantic web search. In *Computational Linguistics and Intelligent Text Processing - 13th International Conference, CICLing 2012, Complementary Proceedings* (New Delhi, India, March 11-17 2012), A. F. Gelbukh, Ed., vol. 2012.

[89] Melo, D., Rodrigues, I. P., and Nogueira, V. B. Work out the semantic web search: The cooperative way. *Advances in Artificial Intelligence 2012* (2012).

[90] Melo, D., Rodrigues, I. P., and Nogueira, V. B. A review on cooperative question answering systems. In *Actas das 3.ª Jornadas de Informática da Universidade de Évora - JIUE'2013* (Departamento de Informática, Escola de Ciências e Tecnologia da Universidade de Évora, February, 21-22 2013), L. Ferreira and V. Pedro, Eds., Escola de Ciências e Tecnologia, Universidade de Évora, pp. 23–30.

[91] Miller, E. An introduction to the resource description framework. *D-Lib Magazine 4*, 5 (1998).

[92] Möller, R., and Haarslev, V. The description logic handbook. Cambridge University Press, New York, NY, USA, 2003, ch. Description logic systems, pp. 282–305.

[93] Nardi, D., and Brachman, R. J. An introduction to description logics. In *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, New York, NY, USA, 2003, pp. 1–40.

[94] Niemelä, I. A tableau calculus for minimal model reasoning. In *Proceedings of the 5th International Workshop on Theorem Proving with Analytic Tableaux and Related Methods* (London, UK, 1996), Springer-Verlag, pp. 278–294.

[95] Oh, J.-H., Torisawa, K., Hashimoto, C., Kawada, T., Saeger, S. D., Kazama, J., and Wang, Y. Why question answering using sentiment analysis and word classes. In *EMNLP-CoNLL* (2012), ACL, pp. 368–378.

[96] Patel-Schneider, P. F., and Horrocks, I. Dlp and fact. In *Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods* (London, UK, 1999), TABLEAUX '99, Springer-Verlag, pp. 19–23.

[97] Perera, R. Ipedagogy: Question answering system based on web information clustering. *2012 IEEE Fourth International Conference on Technology for Education 0* (2012), 245–246.

[98] Pomerantz, J. A linguistic analysis of question taxonomies. *Journal of the American Society for Information Science and Technology (JASIST) 56*, 7 (2005), 715–728.

[99] Quaresma, P., Rodrigues, I., Prolo, C., and Vieira, R. Um sistema de Pergunta-Resposta para uma base de Documentos. *Letras de Hoje 41*, 2 (2006), 43–63.

[100] Quinlan, J. R. Induction of decision trees. *Mach. Learn. 1*, 1 (Mar. 1986), 81–106.

[101] Quintano, L., and Rodrigues, I. Using a logic programming framework to control database query dialogues in natural language. In *Proceedings of the 22nd international conference on Logic Programming* (Berlin, Heidelberg, 2006), ICLP'06, Springer-Verlag, pp. 406–420.

[102] Saint-Dizier, P., and Moens, M.-F. Knowledge and reasoning for question answering: Research perspectives. *Information Processing and Management 47*, 6 (2011), 899–906.

[103] Sattler, U., Calvanese, D., and Molitor, R. Relationship with other formalisms. In *The Description Logic Handbook: Theory, Implementation and Applications*, F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, Eds. Cambridge University Press, 2003, pp. 137–177.

[104] Small, S., and Strzalkowski, T. Hitiqa: High-quality intelligence through interactive question answering. *Natural Language Engineering: Special Issue on Interactive Question Answering 15*, 1 (January 2009), 31–54.

[105] Suchanek, F. M., Kasneci, G., and Weikum, G. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web* (New York, NY, USA, 2007), WWW '07, ACM, pp. 697–706.

[106] Tablan, V., Damljanovic, D., and Bontcheva, K. A natural language query interface to structured information. In *Proceedings of the 5th European semantic web conference on The semantic web: research and applications* (Berlin, Heidelberg, 2008), ESWC'08, Springer-Verlag, pp. 361–375.

[107] Vassiliadis, V., Wielemaker, J., and Mungall, C. Processing OWL2 ontologies using Thea: An application of logic programming. In *OWLED, CEUR Workshop Proceedings* (2009), R. Hoekstra and P. F. Patel-Schneider, Eds., vol. 529, Citeseer.

[108] Voorhees, E. M. Overview of the trec-9 question answering track. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)* (2000), pp. 71–80.

[109] Wang, C., Xiong, M., Zhou, Q., and Yu, Y. Panto: A portable natural language interface to ontologies. In *Proceedings of the 4th European conference on The Semantic Web: Research and Applications* (Berlin, Heidelberg, 2007), ESWC '07, Springer-Verlag, pp. 473–487.

[110] Webb, N., and Webber, B. L. Special issue on interactive question answering: Introduction. *Natural Language Engineering 15*, 1 (2009), 1–8.

[111] Wielemaker, J. An optimised Semantic Web query language implementation in Prolog. *Logic Programming* (2005), 128–142.

[112] Wielemaker, J., Hildebrand, M., and van Ossenbruggen, J. Using Prolog as the fundament for applications on the semantic web. *Proceedings of ALPSWS2007* (2007), 84–98.

[113] Witzig, S. Accessing wordnet from prolog. In *ProNTo - Prolog Natural Language Tools*. Center of Artificial Intelligence, University of Georgia, 2003.

Anexos

Anexo A

Predicados Prolog do Sistema Cooperativo de Pergunta-Resposta para a Web Semântica

Neste Apêndice apresentamos a implementação do conjunto de predicados Prolog que foram introduzidos no Capítulo 6 e que fazem parte fundamental da implementação do sistema Cooperativo de Pergunta-Resposta para a Web Semântica, presente neste trabalho.

A.1 Predicados Prolog que fazem parte do módulo Contexto Ontológico

resources(-URIList, +Name)

`URIList` é a lista de resultados, com a informação associada a cada recurso da DBpedia, que foi devolvido pela ferramenta de pesquisa, e que estão relacionados com o termo `Name`.

```
% resources(-URIList, +Name)
% URIList lista de resultados, com a informacao associada a
    cada recurso da DBpedia, que foi devolvido pela
    ferramenta de pesquisa, e que estao relacionados com o
    termo Name
resources(URIList, Name) :-
  atomic_list_concat(['http://lookup.dbpedia.org/api/search.
```

```
        asmx/PrefixSearch?QueryClass=&MaxHits=10&QueryString=',
         Name], Url),
    http_open(Url, In, []),
    load_structure(In, [element('ArrayOfResult', _, URIList)],
         []),
    close(In).
```

name(-URI, +Name)

`URI` é a URI do recurso da DBpedia que está relacionado com o termo `Name`.

```
% name(-URI, +Name)
% URI recurso da DBpedia que esta relacionado com o termo
    Name
name(URI, Name) :-
    resources(A, Name),
    www_form_encode(Name, FormEncodedName),
    resources(A,FormEncodedName),
    member(element('Result', [], B), A),
    member(element('Result', [], B), A),
    member(element('URI', [], [URIName]), B),
    www_form_encode(URI, URIName).
```

property(-Property, +Term)

`Property` é a URI da propriedade que está relacionada com o termo linguístico `Term`.

```
% property(-Property, +Term)
% Property e uma propriedade da ontologia que representa
    Term
property(Property, Term) :-
    property(Property), downcase_atom(Property, P),
        downcase_atom(Term, T),
    in_string(P, T).

property(Property, Term) :-
    atomic_list_concat(['SELECT ?p WHERE { ?p rdf:type rdf:
        Property . ?p rdfs:label ?y . ?y bif:contains "', Term,
        '" }'], Query),
    sparql_query(Query, row(Property), [ host('dbpedia.org'),
         path('/sparql/') ]).
```

property(-Property, -Class, +Term)

`Property` é a URI da propriedade que está relacionada com o termo linguístico `Term`. A propriedade `Property` é um comportamento definido na classe `Class`

```
% property(-Property, -Class, +Term)
% Property e um comportamento definido na classe Class
% Property e uma propriedade da ontologia que representa
    Term
property(Property, Class, Term) :-
  (propertyDomain(Property, Class); propertyRange(Property,
      Class)),
  atomic_list_concat(['SELECT ?x WHERE { <', Property, '>
      rdfs:label  ?x }'], Query),
  sparql_query(Query, row(literal(lang(en, Y))), [ host('
      dbpedia.org'), path('/sparql/') ]),
  in_string(Y, Term).
```

synonymous(+Word, -Category, -Synonymous)

`Synonymous` é o synset do termo `Word`, obtido da WordNet e que pertence à categoria `Category`.

```
% synonymous(+Word, -Category, -Synonymous)
% sinonimos
% Categoria pode ser n(oun), v(erb), a(djective), s(atellite
    adjective), r(adverb)
synonymous(Word, Category, Synonymous) :-
        s(Word_Synset, Word_Count, Word, Category, _, _),
        sin_aux(Word_Synset, Word, Synonymous, Category).

% mesmo synset
sin_aux(Word_Synset, Word, Synonymous, Categ) :-
        member_synset(Word_Synset, Synonymous, Category),
        Word \= Synonymous.

% hyp
sin_aux(Word_Synset, Word, Synonymous, Category) :-
        hyp(Word_Synset, Sin_Synset),
        member_synset(Sin_Synset, Synonymous, Category).

% derivationally related form
sin_aux(Word_Synset, Word, Synonymous, Category) :-
        der(Word_Synset, _, Sin_Synset, _),
        member_synset(Sin_Synset, Synonymous, Category).


member_synset(Word_Synset, Word, Category) :-
        s(Word_Synset, _, Word, Categ, _, _).
```

A.2 Predicados Prolog que fazem parte do módulo Avaliação Semântica

query(+Y, +Property, -X)

Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `X` que verificam a consulta. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação inversa.

```
% query(+Y, +Property, -X)
% o recurso X (objeto do triplo) tem valor Y (sujeito do
    triplo) para a propriedade Property (predicado do triplo)
query(Y, Property, X) :-
  Y \= null, Property \= null,
  in_string(Y, 'http://'), in_string(Property, 'http://') ->
     atomic_list_concat(['SELECT * { ?x <', Property, '> <',
         Y, '> }'], Query),
     sparql_query(Query, row(X), [ host('dbpedia.org'),
        path('/sparql/') ]).
```

query(+X, +Property, -Y)

Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `Y` que verificam a consulta. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação direta.

```
% query(+X, +Property, -Y)
% o recurso X (sujeito do triplo) tem valor Y (objeto do
    triplo) para a propriedade Property (predicado do triplo)
query(X, Property, Y) :-
  X \= null, Property \= null,
  in_string(X, 'http://'), in_string(Property, 'http://') ->
     atomic_list_concat(['SELECT * { <', X, '> <', Property,
         '> ?y }'], Query),
     sparql_query(Query, row(literal(lang(en,Y))), [ host(
        'dbpedia.org'), path('/sparql/') ]).

% query(+X, +Property, -Y)
% o recurso X (sujeito do triplo) tem valor Y (objeto do
    triplo) para a propriedade Property (predicado do triplo)
query(X, Property, Y) :-
  X \= null, Property \= null,
  in_string(X, 'http://'), in_string(Property, 'http://') ->
```

```
        atomic_list_concat(['SELECT * { <', X, '> <', Property,
            '> ?y }'], Query),
        sparql_query(Query, row(Y), [ host('dbpedia.org'),
           path('/sparql/') ]).
```

query(+X, +Property, +Term, -Y)

Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `Y` que verificam a consulta, quando estão relacionadas com o termo `Term`. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação direta.

```
% query(+X, +Property, +Term, -Y)
% o recurso X (sujeito do triplo) tem valor Y (objeto do
    triplo que representa Term) para a propriedade Property (
    predicado do triplo)
query(X, Property, Term, Y) :-
  X \= null, Property \= null,
  in_string(X, 'http://'), in_string(Property, 'http://') ->
      atomic_list_concat(['SELECT ?y { <', X, '> <', Property,
          '> ?y . FILTER regex(?y, "', Term, '") }'], Query),
      sparql_query(Query, row(Y), [ host('dbpedia.org'),
          path('/sparql/') ]).
```

query(-X, +Property, +Term, -Y)

Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `X, Y` que verificam a consulta, quando as entidades `Y` estão relacionadas com o termo `Term`. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação direta.

```
% query(-X, +Property, +Term, -Y)
% o recurso X tem valor Y (que representa Term) para a
    propriedade Property
query(X, Property, Term, Y) :-
  Property \= null,
  in_string(Property, 'http://') ->
      atomic_list_concat(['SELECT ?x ?y { ?x <', Property, '>
          ?y . FILTER regex(?y, "', Term, '") }'], Query),
      sparql_query(Query, row(X, Y), [ host('dbpedia.org'),
          path('/sparql/') ]).
```

query(+X, -Property, +Term, -Y)

Interpreta um predicado da DRS como consulta SPARQL, acedendo à base de dados da DBpedia e retornando as entidades `P, Y` que verificam a consulta, quando as entidades `Y` estão relacionadas com o termo `Term`. `X` representa o sujeito, `Property` representa o predicado e `Y` representa o objeto traduzidos em triplos RDF. O triplo RDF é interpretado como relação direta.

```
% query(+X, -Property, +Term, -Y)
% o recurso X (sujeito do triplo) tem valor Term (objeto do
    triplo) para a propriedade Property (predicado do triplo)
query(+X, -P, +Term, -Y) :-
  X \= null,
  in_string(X, 'http://') ->
     atomic_list_concat(['SELECT ?p ?y { <', X, '> ?p ?y . .
        FILTER regex(?y, "', Term, '") }'], Query),
     sparql_query(Query , row(P, Y), [ host('dbpedia.org'),
         path('/sparql/') ]).
```

isa(-X,+Y)

Verifica/estabelece a semelhança entre `X` e `Y`. No caso mais simples permite verificar ou estabelecer que `X = Y`.

```
% isa(-X,Y+)
% verifica se duas instancias estao relacionadas
isa(literal(lang(en,X)),literal(lang(en,X))) :- !.
isa(literal(type(X,Y)),literal(type(X,Y))) :- !.
isa(X,X).
isa(X,Y) :- !, name(X,Y).
isa(X,Y) :- !, classOf(X,Y).
isa(X,Y) :- !, property(X,Y).
```

add_to_context(+Data)

O facto `Data` é adicionado ao contexto do discurso da questão. Este predicado recorre ao predicado pré-definido `assert(+Term)` (a expressão `Term` é definida como a última cláusula ou facto do predicado).

```
%  add_to_context(+Data)
%  adicionar o facto Data ao contexto da questao
add_to_context(Data) :-
  (exist(Data) -> ( Data -> writeln('Regra existe')
                         ; (newRule(Data)))
                ; (newRule(Data)) ),
  append('Context.pl'),       % open this file
  listing(Data),              % list all clauses in memory
  told.
```

```
% newRule(+Rule)
% define o predicado Rule
newRule(A) :- assert(A).

% exist(+Rule)
% verifica se existe o predicado Rule definido
exist(Rule) :-
  Rule =.. [Name, Arg1],
  current_predicate(Name/1).

% exist(+Rule)
% verifica se existe o predicado Rule definido
exist(Rule) :-
  Rule =.. [Name, Arg1, Arg2],
  current_predicate(Name/2).
```

A.3 Predicados Prolog que fazem parte do módulo Controlador do Discurso

presuppositions_solutions(+Variables, +Presuppositions_List, -Presuppositions_Solutions_List)

Extrai a lista de entidades da ontologia, valores das variáveis, que verificam os predicados pressupostos que constituem a DRS da questão. `Varibles` é a lista de variáveis associados aos referentes da DRS da questão, `Presuppositions_List` é a lista de predicados pressupostos da DRS da questão e `Presuppositions_Solutions_List` é a lista de soluções, valores das variáveis, que verificam os pressupostos.

```
% presuppositions_solutions(+Variables, +
    Presuppositions_List, -Presuppositions_Solutions_List)
% Extrai a lista de entidades que verificam os pressupostos
presuppositions_solutions(Variables, Presuppositions_List,
    Presuppositions_Solutions_List) :-
  findall(Variables, process_list(Presuppositions_List),
      Presuppositions_Solutions_List).
```

main_solution(+Variables, +Presuppositions_Solutions_List, +Main_Predicates_List, -Main_Solutions_List)

Extrai da lista de entidades da ontologia, que são solução dos predicados pressupostos, as entidades que verificam os predicados principais que constituem a DRS da questão. `Varibles` é a lista de variáveis associados aos referentes da DRS da

questão, `Presuppositions_Solutions_List` é a lista de entidades que são solução dos predicados pressupostos da DRS da questão, `Main_Predicates_List` é a lista de predicados principais da DRS da questão e `Main_Solutions_List` é a lista de soluções, valores das variáveis, dos predicados pressupostos que verificam os predicados principais.

```
% main_solution(+Variables, +Presuppositions_Solutions_List,
    +Main_Predicates_List, -Main_Solutions_List)
% extrai a lista de entidades - solucoes dos pressupostos -
   que verificam os predicados principais
main_solutions(Variables, Presuppositions_Solutions_List,
   Main_Predicates_List, Main_Solutions_List) :-
  findall(Variables, (member(Variables,
     Presuppositions_Solutions_List), solution(Variables,
     Main_Predicates_List)), Main_Solutions_List).
```

get_solution(-Variables_Values, +Predicates_List)

Extrai as soluções que verificam os predicados. `Variables_Values` é o valor das variáveis associados aos referentes da DRS da questão e `Predicates_List` é a lista de predicados que têm de ser verificados.

```
% get_solution(-Variables_Values, +Predicates_List)
% extrai valores das variaveis que verificam predicados
get_solution(Variables_Values, Predicates_List) :-
  process_list(Predicates_List), !.
```

process_list(+Predicates_List)

Processa a lista de predicados `Predicates_List` que constituem a DRS da questão, i.é. invoca e controla a execução da Descoberta da Ontologia e a Avaliação Semântica.

```
% process_list(+Predicates_List)
% processa a lista de predicados
% procura os termos associados aos predicados na ontologia
% valida os termos: extraindo/verificando as entidades/
   termos na ontologia
process_list([]).
process_list([date(X)|Xs]) :-
  !, date(X), process_list(Xs).
process_list([isa(X, Y)|Xs]) :-
  !, isa(X,Y), process_list(Xs).
process_list([was(X, Y)|Xs]) :-
  !, was(X,Y), process_list(Xs).
process_list([name(URI, Name)|Xs]) :-
  !, name(URI, Name), process_list(Xs).
process_list([Term_Args|Xs]) :-
```

```
    Term_Args =.. [Term, X], find(Term, X), process_list(Xs).
process_list([Term_Args|Xs]) :-
    Termo_Args =.. [Term, X, Y],
    (find(Term, X, Y); find(Term, Y, X)),
    process_list(Xs).
```

find(+Term, -X) e find(+Term,-X,-Y)

Fazem o mapeamento dos predicados linguísticos da DRS da questão nos termos da ontologia, bem como a sua validação na ontologia. O predicado Prolog `find/2` corresponde aos predicados linguísticos com apenas um argumento `X`, onde `Term` corresponde ao nome do predicado. O predicado Prolog `find/3` corresponde aos predicados linguísticos com dois argumentos `X` e `Y`, onde `Term` corresponde ao nome do predicado.

```
find(Term, literal(type(_,X))) :-
    classOf(URI_Class, Term), instance(X, URI_Class).
find(Term, literal(type(_,X))) :-
    name(X, Term).
find(Term, literal(type(_,X))) :-
    name(Term_URI, Term), query(X, P_URI, Termo_URI).
find(Term, literal(type(_,X))) :-
    query(X, P_URI, Y, Term).
find(Term, literal(type(_,X))) :-
    !, instance(X, Class),
    property(Property, Class, Term),
    query(X, Property, Valor).
find(Term, literal(lang(_,X))) :-
    classOf(URI_Class, Term), instance(X, URI_Class).
find(Term, literal(lang(_,X))) :-
    name(X, Term).
find(Term, literal(lang(_,X))) :-
    name(Term_URI, Term), query(X, P_URI, Term_URI).
find(Term, literal(lang(_,X))) :-
    query(X, P_URI, Y, Term).
find(Term, literal(lang(_,X))) :-
    !, instance(X, Class),
    property(Property, Class, Term),
    query(X, Property, Valor).

% Term e uma classe? X instancia da classe?
find(Term, X) :-
    classOf(URI_Class, Termo), instance(X, URI_Class).

% Term e um recurso? X = URI_Termo
```

```
find(Term, X) :-
  name(X, Term).

% Termo e um recurso? X tem como caracteristica o Atributo?
find(Term, X) :-
  name(Term_URI, Term), query(X, P_URI, Term_URI).

% Termo e um atributo? X tem como caracteristica o Termo?
find(Term, X) :-
  query(X, P_URI, Y, Term).

% Atributo e uma propriedade? X e uma instancia que verifica
    a propriedade
find(Term, X) :-
  instance(X, Class),
  property(Property, Class, Term),
  query(X, Property, Value).

% Termo e uma propriedade? X termo Y
find(Term, X, Y) :-
  (classOf(Class, X); classOf(Class, Y)),
  property(Property, Class, Term),
  query(X, Property, Y).

% Termo e uma propriedade? Y termo X
find(Term, Y, X) :-
  (classOf(Class, X); classOf(Class, Y)),
  property(Property, Class, Term),
  query(Y, Property, X).

% Termo e uma propriedade? pretendemos encontar Y: X Termo Y
find(Term, X, Y) :-
  property(Property, Term),
  query(X, Property, Y).

% Termo e uma propriedade? segundo argumento e uma palavra,
   X termo Word
find(Term, X, Word) :-
  Word \= null ->
     property(Property, Term), query(X, Property, Word,
        Value).
```

instance(-X, +Class)

Verifica/determina se a entidade `X` é uma instância da classe da ontologia `Class`.

```
% instance(-X, +Class)
% determina/verifca se X e instancia da classe Class
% todos os recursos sao things (classe principal)
instance(_, 'http://dbpedia.org/ontology/Thing').
instance(X, Class):-
  Class \= null, in_string(Class, 'http://') ->
      atomic_list_concat(['SELECT ?instance WHERE { ?instance
          <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
          <', Class, '>}'], Query),
      %sparql_query(Query , row(X), [ host('dbpedia.org'),
          path('/sparql/') ]).
      sparql_query(Query , row(X), [ host('localhost'), port
          (8890), path('/sparql/') ]).

instance(X, Class):-
  X \= null, in_string(X, 'http://') ->
      atomic_list_concat(['SELECT ?class WHERE {<', X, '><
          http://www.w3.org/1999/02/22-rdf-syntax-ns#type> ?
          class }'], Query),
      %sparql_query(Query , row(Class), [ host('dbpedia.org
          '), path('/sparql/') ]).
      sparql_query(Query , row(Class), [ host('localhost'),
          port(8890), path('/sparql/') ]).
```

classOf(-Class, +X)

Determina/verifica se a classe `Class` está relacionada com o termo `X`. Se o termo `X` é um termo linguístico então a classe `Class` é o mapeamento na ontologia do termo. Se o termo `X` é uma entidade da ontologia então verifica/determina que o termo é uma instância da classe `Class`.

```
% definicao da classe principal case sensitive
classOf('http://dbpedia.org/ontology/Thing', 'Thing').
classOf('http://dbpedia.org/ontology/Thing', 'thing').
% Class e uma classe da ontologia que representa Term
classOf(Class, Term) :-
  Term \= null,
  class(Class),
  ((annotationAssertion('http://www.w3.org/2000/01/rdf-
      schema#label',Class,literal(lang(en,Y))),
  in_string(Y, Term));
  downcase_atom(Class, Class1), downcase_atom(Term, Term1),
  in_string(Class1, Term1)).
```

```
% Class e uma classe da dbpedia que tem X como instancia
classOf(Class, X):-
  X \= null, in_string(X, 'http://') ->
      atomic_list_concat(['SELECT ?class WHERE { <', X, '> <
          http://www.w3.org/1999/02/22-rdf-syntax-ns#type> ?
          class }'], Query),
      sparql_query(Query , row(Class), [ host('dbpedia.org')
          , path('/sparql/') ]),
      class(Class).
```

get_properties(+Referent_list_values)

Coleciona as propriedades da ontologia que estão relacionada com os valores dos referentes contidos na lista `Referent_list_values` e adicona cada uma delas ao contexto da questão.

```
% get_properties(Referent\_list\_values)
% coleciona todas as propriedades associadas aos valores dos
    da lista Referent_list_values
get_properties(Referent_list_values) :-
  aggregate_all( bag(Prop), ( member(Elem,
      Referent_list_values),
                              atomic_list_concat(['SELECT ?p
                                  ?value { <', Elem, '> ?p ?
                                  value . }'], Query),
                              sparql_query(Query , row(P,
                                  Value), [ host('dbpedia.
                                  org'), path('/sparql/') ]),
                                (triple(Elem,P,Value) ->
                                    true;
                                (add_to_context(
                                    prop_solution(Elem,P,
                                    Value)), Prop =
                                    prop_solution(Elem,P,
                                    Value))),
                              add_to_context(prop_solution(
                                  Elem,P,Value)), Prop =
                                  prop_solution(Elem,P,Value)
                                  ),
                    Propriedades
                  ).
```

data_entropy(+T, +Referent_list_values, -Value)

`Value` é o valor da entropia do conjunto de dados T, relativamente às classes dos dados pertencentes à lista `Referent_list_values`.

```
% data_entropy(+T, +Referent_list_values, -Value)
% T conjunto das propriedades - cada elemento e da forma
    prop_solucao(X,P,Value)
% soma dos -pi*log(pi;2), onde i elemento solucao
% pi = frequencia absoluta de i em S / total de elementos de
    S
data_entropy(T,Referent_list_values,Value) :-
  length(T,TotalT),
  aggregate_all(sum(Ent), ( member(Elem,
      Referent_list_values),
                            aggregate_all(count, prop_solucao(
                               Elem,_,_), Num),
                            Prob is (Num/TotalT),
                            Ent is (-Prob * log(Prob) / log(2))
                               ),
                Value).
```

property_entropy(+P, +Referent_list_values, -EntropyValue)

`EntropyValue` é o valor da entropia da propriedade `P`, para as classes `Referent_list_values`.

```
% property_entropy(+P, +Referent_list_values, -EntropyValue)
% calcula a entropia da propriedade P para as classes na
    lista Referent_list_values
% T - conjunto dos triplos que contem a propriedade P
property_entropy(P, Referent_list_values, EntropyValue) :-
  aggregate_all(bag(prop_solution(X,P,V)), prop_solution(X,P
      ,V), T),
  length(T, TotalT),
  sum_entropy_prop_value(T,TotalS,P,Referent_list_values,
      EntropyValue).


sum_entropy_prop_value([],_,_,_,0).
sum_entropy_prop_value(S,TotalS,P,Referent_list_values,
    EntropyValue) :-
  member(prop_solution(_,_,Value),S),
  aggregate_all(bag(prop_solution(X,P,Value)), member(
      prop_solution(X,P,Value), S), S1),
  %retirar os elementos prop_solution(_,P,Value) de S
  aggregate_all(bag(prop_solution(A,P,V)),(member(
      prop_solution(A,P,V),S), not(member(prop_solution(A,P,V
```

```
    ),S1))), NewS),
  sum_entropy_prop_value(NewS, TotalS,P, Referent_list_values,
     NewEnt),
  entropy_propertie_value(Value, S1, Referent_list_values,
     EntropyPValue),
  length(S1, TotalS1),
  Prob is TotalS1 / TotalS,
  (Prob \= 0 -> Ent is (Prob * EntropyPValue); Ent is 0),
  EntropyValue is NovoEnt + Ent.


entropy_property_value(Value, S, Referent_list_values,
   EntropyPValue) :-
  length(S, TotalS),
  aggregate_all(sum(Ent), ( member(Elem,
     Referent_list_values),
                            aggregate_all(count, member(
                               prop_solution(Elem,_,_), S), Num
                               ),
                            Prob is (Num/TotalS),
                            (Prob \= 0 -> Ent is (-Prob * log(
                               Prob) / log(2)); Ent is 0) ),
                EntropyPValue), !.
```

gain_property(S, +Prop, +Referent_list_values, -Gain)

`Gain` e o ganho de informação da propriedade `P` no conjunto de dados `S` e para as classes da lista `Referent_list_values`.

```
% gain_property(S, +Prop, +Referent_list_values, -Gain)
% Gain e o ganho de informacao da propriedade P no conjunto
   de dados S e para as classes da lista
   Referent_list_values
gain_property(S,Prop,Referent_list_values,Gain) :-
  data_entropy(S,Referent_list_values, EntropyS), !,
  property_entropy(Prop, Referent_list_values, EntropyP),
  Gain is EntropyS - EntropyP, !.
```

ratio_gain_property(+Property_entropy, +Gain_property, -Ratio)

`Ratio` e a razão do ganho de informação, com `Property_entropy` o valor da entropia da propriedade e `Gain_property` o ganho de informação da mesma propriedade.

```
% ratio_gain_property(+Property_entropy, +Gain_property, -
   Ratio)
```

```
% Ratio e a razao do ganho de informacao, com
    Property_entropy o valor da entropia da propriedade e
    Gain_property o ganho de informacao da mesma propriedade
ratio_gain_property(Property_entropy, Gain_property, Ratio)
    :-
  (Property_entropy \= 0.0 -> Ratio is (Gain_property /
     Property_entropy); Ratio = 'NA').
```