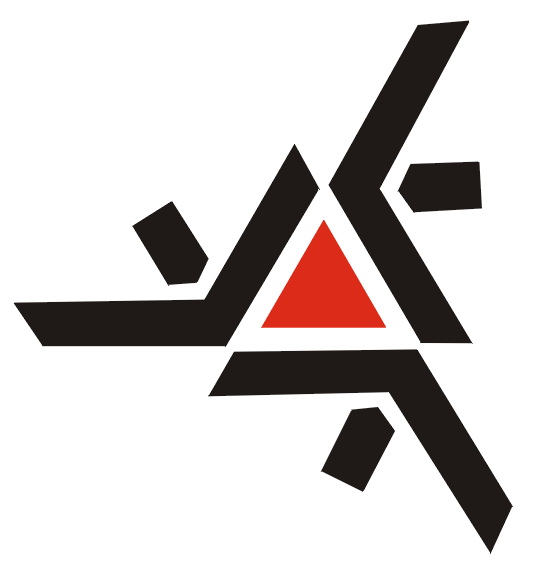
**Universidade Estadual de Maringá**

**Centro de Tecnologia - Departamento de Informática**

**Bacharelado em Ciência da Computação**

**Trabalho De Conclusão De Curso II - 2016**

gabriel vinicius papa belini

aprimoramento de buscas na internet por meio da criação de uma ontologia para a área de inteligência artificial

MARINGÁ

2016

**GABRIEL VINICIUS PAPA BELINI**

Aprimoramento de Buscas na Internet por meio da Criação de uma Ontologia para a Área de Inteligência Artificial

Trabalho de Conclusão de Curso de Graduação apresentado ao Departamento de Informática da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

**Orientador: Prof. Wagner Igarashi, Dr.**

MARINGÁ

2016

Aprimoramento de Buscas na Internet por meio da Criação de uma Ontologia para a Área de Inteligência Artificial

Gabriel Vinicius Papa Belini

Trabalho de Conclusão de Curso de Graduação apresentado ao Departamento de Informática da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Wagner Igarashi, Dr.

UEM (Universidade Estadual de Maringá)

(Orientador)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. André Barbosa Verona, M.e

UEM (Universidade Estadual de Maringá)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Profa. Valéria Delisandra Feltrim, Dra.

UEM (Universidade Estadual de Maringá)

**DEDICATÓRIA**

À toda minha família, em especial, minha mãe Rozilene, meu pai Aparecido e minha irmã Nathália.

**AGRADECIMENTOS**

AOS MEUS PAIS, que sempre me apoiaram e me incentivaram a buscar uma formação acadêmica, mesmo nos momentos mais difíceis me deram forças e não me deixaram desistir do curso ao longo dos anos.

À MINHA IRMÃ NATHÁLIA, sempre de bom humor e muito otimista me ajudou a aprender a estudar e sempre foi um exemplo a ser seguido. Além de ser um motivo de orgulho por ter coragem de abandonar um curso de graduação para tentar outro muito mais difícil, é um motivo de inspiração pelas conquistas obtidas e pelo exemplo de perseverança.

À MINHA NAMORADA ANDRESSA, que sempre me ajudou dando conselhos e quando passei pelo momento mais difícil do curso foi ela que conseguiu me acalmar e me conscientizar de que meu esforço seria válido e recompensado.

AO MEU TIO RONALD, que sempre foi um exemplo de pessoa culta e dedicada a enriquecer seu conhecimento cada vez mais. Sempre me apoiou e me deu forças para continuar lutando por uma formação acadêmica.

AOS MEUS AMIGOS, todos os que acreditaram em mim, em especial ao Carlos Paísca que sempre me ajudou a me organizar, estudar e me tornar cada dia mais responsável a exemplo dele, e também ao Lucas Helal, Gustavo Borelli e Juliano Donini por estarem sempre presentes nos momentos bons e ruins.

AO MEU ORIENTADOR, PROF. DR. WAGNER IGARASHI, sem ele esse trabalho não seria concluído. Agradeço por sempre conseguir esclarecer as ideias e contribuir positivamente com o trabalho, fazendo críticas construtivas e sanando todas as dúvidas que surgiam durante o caminho.

*“A year from now, five years from now, what you gonna wish you did today?”*

*(Autor Desconhecido)*

**RESUMO**

Com o crescimento desenfreado de informações no meio digital, torna-se cada vez mais imprescindível o uso de mecanismos específicos para classificar esses dados e facilitar a recuperação de informação. Com a utilização e criação de ontologias diversos sistemas de busca permitem que o usuário efetue a recuperação de informação utilizando a semântica da palavra, reduzindo assim o escopo da busca e aumentando a precisão.

Neste contexto, o trabalho realizado tem como objetivo construir uma ontologia para a área de inteligência artificial, classificando seus principais termos de forma hierárquica e por fim fazer o uso dessa ontologia em um protótipo, o qual permitirá que o usuário efetue buscas em sites de busca utilizando a ontologia desenvolvida.

**Palavras-chave:** Ontologia. Recuperação de Informação. Inteligência Artificial. Busca semântica.

**ABSTRACT**

With the uncontrolled growth of information in the digital environment, it becomes more and more necessary the use of specific mechanisms to classify such data and facilitate information retrieval. With the use and creation of ontologies many search systems allows the user to make information retrieval using the word semantics, thereby reducing the scope of the search and increasing the accuracy. In this sense , the work aims to build an ontology for the field of Artificial Intelligence , classifying its principal terms hierarchically in order to make use of this ontology in a prototype , which allows the user to perform searches on search engines using the ontology developed.

**Keywords:** Ontology. Information Retrieval. Artificial Intelligence. Semantic Search.

**LISTA DE ILUSTRAÇÕES**

[Figura 1: Ordem com a qual o algoritmo DFS percorre o grafo. 23](#_Toc457304290)

[Figura 2: Ordem com a qual o algoritmo BFS percorre o grafo. 23](#_Toc457304291)

[Figura 3: Ordem de visitação dos nós utilizando o algoritmo de Busca pelo Custo Uniforme. 24](#_Toc457304292)

[Figura 4: Principais processos do método de Uschold e King. 28](#_Toc457304293)

[Figura 5: Interface do software Protégé mostrando classes e subclasses da ontologia. 36](#_Toc457304294)

[Figura 6: Trecho de código em Python, mostrando utilização da biblioteca ontospy. 37](#_Toc457304295)

[Figura 7: Resultado da função printClassTree() da biblioteca ontospy. 38](#_Toc457304296)

[Figura 8: Caso de uso do protótipo desenvolvido. 39](#_Toc457304297)

[Figura 9: Imagem ilustrando a interface principal do protótipo. 40](#_Toc457304298)

[Figura 10: Resultado apresentado pelo protótipo após efetuar a busca. 41](#_Toc457304299)

[Figura 11: Trecho de código em Python, mostrando a função adicionaOutros() 42](#_Toc457304300)

[Figura 12: Trecho de código em Python mostrando as funções da biblioteca Selenium utilizadas. 43](#_Toc457304301)

**LISTA DE QUADROS**

Quadro 1: Parte da ontologia modelada utilizando Excel..................................32

**LISTA DE ABREVIATURES E SIGLAS**

ACM Association for Computing Machinery

API Application Programming Interface

BFS Breadth First Search

CCS Computing Classification System

DFS Depth First Search

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

IA Inteligência Artificial

IDE Integrated Development Environment

OWL Web Ontology Language

PHP Hypertext Preprocessor

PLN Processamento de Linguagem Natural

RDF Resource Description Framework

RI Recuperação de Informação

W3C World Wide Web Consortium

XML Extensible Markup Language

**SUMÁRIO**

[1. INTRODUÇÃO 13](#_Toc457304148)

[1.1 OBJETIVOS 14](#_Toc457304149)

[1.2 DESCRIÇÃO DA PROBLEMÁTICA 15](#_Toc457304150)

[1.3 JUSTIFICATIVA 15](#_Toc457304151)

[1.4 MATERIAIS E MÉTODOS 15](#_Toc457304152)

[1.5 RESULTADO OBTIDO 16](#_Toc457304153)

[1.6 ORGANIZAÇÃO DO TRABALHO 16](#_Toc457304154)

[2. FUNDAMENTAÇÃO TEÓRICA 18](#_Toc457304155)

[2.1 INTELIGÊNCIA ARTIFICIAL 18](#_Toc457304156)

[2.1.1 DEFINIÇÕES 18](#_Toc457304157)

[2.1.2 SUBÁREAS DA IA 19](#_Toc457304158)

[2.1.2.1 REPRESENTAÇÃO DE CONHECIMENTO 19](#_Toc457304159)

[2.1.2.2 PROCESSAMENTO DE LINGUAGEM NATURAL (PLN) 20](#_Toc457304160)

[2.1.2.3 APRENDIZAGEM DE MÁQUINA 21](#_Toc457304161)

[2.1.2.4 ALGORITMOS 21](#_Toc457304162)

[2.2 ONTOLOGIA 25](#_Toc457304163)

[2.2.1 FERRAMENTAS DISPONÍVEIS 26](#_Toc457304164)

[2.2.2 FORMAS DE MODELAGEM DE UMA ONTOLOGIA 28](#_Toc457304165)

[2.3 TRABALHOS CORRELATOS 29](#_Toc457304166)

[3. DESENVOLVIMENTO 32](#_Toc457304167)

[3.1 ESTUDO DE FERRAMENTAS ESPECÍFICAS PARA O DESENVOLVIMENTO DE ONTOLOGIAS 32](#_Toc457304168)

[3.2 CRIAÇÃO DA ONTOLOGIA 32](#_Toc457304169)

[3.3 IMPLEMENTAÇÃO DE UM SCRIPT PARA PROCESSAR O ARQUIVO .OWL 36](#_Toc457304170)

[3.4 IMPLEMENTAÇÃO DO PROTÓTIPO DE BUSCAS AUTOMATIZADAS COM INTERFACE GRÁFICA 38](#_Toc457304171)

[3.4.5 INFORMAÇÕES SOBRE O CÓDIGO IMPLEMENTADO 41](#_Toc457304172)

[4. CONCLUSÕES 44](#_Toc457304173)

[5. REFERÊNCIAS BIBLIOGRÁFICAS 46](#_Toc457304174)

1. INTRODUÇÃO

Globalização é o processo de integração e interação entre pessoas do mundo todo. Uma ferramenta de suma importância para a propagação deste fenômeno é a *Internet,* um meio de comunicação o qual permite que pessoas do mundo inteiro troquem informações e realizem diversas atividades em tempo real.

Com o passar dos anos o acesso à *Internet* tem se tornado cada vez mais fácil, permitindo que cada vez mais pessoas conectem-se e publiquem informações na rede. Esse aumento desenfreado na quantidade de informação disponível na *Internet* trouxe alguns problemas, sendo um deles o de recuperar informações que sejam relevantes aos usuários em pesquisas por palavras chave, por exemplo, pesquisas no Google.

Diversas técnicas são utilizadas por sites de busca na recuperação de informações, porém, em sua maioria, não levam em consideração a semântica das palavras sendo pesquisadas. Essa metodologia muitas vezes não retorna resultados satisfatórios tendo em vista que os resultados terão apenas documentos que contenham as palavras chave utilizadas na busca e caso existam documentos semanticamente relacionados com a palavra chave, mas que não contenham a mesma em seu corpo, tais documentos serão descartados pelos métodos tradicionais.

Na Inteligência Artificial temos o conhecimento como um elemento chave utilizado na resolução de problemas, porém esse conhecimento precisa ser representado de certa forma para que possa ser utilizado. Uma forma de representar conhecimento de maneira formal é por meio do que chamamos de ontologia, que é uma técnica utilizada na representação de conhecimento formal, procurando explicitar formalmente relações semânticas entre conceitos de um mesmo domínio. Uma ontologia é, segundo Gruber (1993), “uma especificação explícita de uma conceituação do domínio”.

Sabendo que ontologias relacionam semanticamente conceitos de domínios específicos e limitados, caso tenhamos uma ontologia definida para um certo domínio podemos montar uma base de conhecimento para auxiliar na recuperação de informação (RI), utilizando junto a outras técnicas a semântica da palavra chave utilizada na busca.

Além disso, as ontologias possuem diversas outras aplicações, dentre elas podemos citar a classificação de textos e artigos científicos, aquisição de conhecimento, reuso do conhecimento e compartilhamento do conhecimento.

Visando melhorar a qualidade da informação retornada ao usuário pelas buscas na Internet, técnicas para a criação de ontologias serão estudadas e aplicadas na criação de uma ontologia específica para a área de Inteligência Artificial, relacionando semanticamente os principais conceitos da área por meio de ferramentas específicas para a criação de ontologias. Tal ontologia poderá ser então testada em conjunto com mecanismos de busca já existentes para que haja uma comparação entre métodos tradicionais e o método aqui proposto.

Nos próximos capítulos veremos um pouco sobre a importância da realização desse trabalho, os materiais e métodos que serão utilizados para o desenvolvimento e também uma visão geral da base teórica já existente sobre o assunto.

1.1 OBJETIVOS

Este trabalho possui como objetivo geral definir uma ontologia para a área de Inteligência Artificial.

Com base no objetivo geral, foram estabelecidos os seguintes objetivos específicos:

* Identificar elementos teóricos necessários para a construção de ontologias;
* Analisar conceitos da área de Inteligência Artificial;
* Identificar estudos correlatos de ontologias em diversas áreas;
* Analisar ferramentas para a criação de ontologias;
* Modelar uma ontologia para conceitos da área de Inteligência Artificial;
* Demonstrar a utilização da ontologia a partir de uma prova de conceito.

1.2 DESCRIÇÃO DA PROBLEMÁTICA

O volume de informação presente na *Internet* tem crescido intensamente nos últimos anos. Isso é decorrente do constante crescimento na facilidade de acesso e no número de pessoas com acesso à *Internet.* Esse grande volume de informação é um obstáculo para os mecanismos de busca como *Google, Yahoo,* etc.

Além da grande quantidade de informação, sites de busca por palavra chave normalmente desconsideram a semântica das palavras buscadas, o que pode causar resultados insatisfatórios.

1.3 JUSTIFICATIVA

O crescimento acelerado da quantidade de informação presente na *Internet* trouxe consigo uma maior dificuldade para os sistemas de recuperação de informação de retornarem resultados satisfatórios, tais sistemas não têm conseguido fazer buscas adequadas com a finalidade que os usuários necessitam por utilizarem apenas palavras chave como busca e desconsiderando a semântica dos termos buscados.

Sendo assim, a modelagem de ontologias pode auxiliar usuários que não têm experiência ou mesmo aumentar o poder de busca ao realizarem consultas na internet.

A realização desse trabalho também irá contribuir para o aprendizado do aluno, visto que o conteúdo abordado não é ministrado nas matérias da graduação além de permitir ao aluno uma experiência com relação à pesquisa cientifica.

1.4 MATERIAIS E MÉTODOS

Uma ontologia é uma espécie de vocabulário sobre conceitos de um domínio específico e que contém relações entre esses diversos conceitos. Será utilizado para a modelagem da ontologia proposta, a ferramenta *open-source* gratuita chamada *protégé* criada na Universidade de Medicina de Stanford, na Califórnia. Essa ferramenta possui uma interface gráfica bastante amigável e permite a criação e edição de ontologias de maneira simples e objetiva.

Para demonstrar a utilização da ontologia será também utilizado a linguagem de programação *Python* em conjunto com a IDE *PyCharm,* distribuida pela *JetBrains,* onde será desenvolvida a aplicação que permitirá a utilização da ontologia criada na recuperação de informação.

No que diz respeito à prova de conceito, sites de busca já existentes serão utilizados em conjunto com a aplicação desenvolvida em *Python,* para que possa ser feita a análise dos resultados. A aplicação funcionará tentando criar uma string de busca baseado na ontologia proposta contendo os conceitos relacionados semânticamente à palavra chave fornecida pelo usuário.

O hardware utilizado no desenvolvimento tanto da ontologia quanto do *script* será um notebook com as seguintes configurações:

* Processador Intel Core i7;
* Memória RAM 8GB DDR3;
* Armazenamento HD 1TB;
* Sistema Operacional Microsoft Windows 10 e Linux Ubuntu 14.04.

1.5 RESULTADO OBTIDO

Como resultado obtido deste trabalho temos uma ontologia para a área de Inteligência Artificial e um protótipo capaz de utilizar dados provindos de ontologias de qualquer área para montar uma interface gráfica e permitir que o usuário a utilize para fazer buscas na internet utilizando como termos as classes definidas na ontologia.

**1.6 ORGANIZAÇÃO DO TRABALHO**

Este trabalho está organizado em cinco seções, a primeira que termina aqui e possui caráter introdutório. A segunda seção busca apresentar uma fundamentação teórica sobre o assunto tratado contando também com uma subseção de trabalhos correlatos. A seção 3 foi utilizada para descrever detalhadamente como foi o desenvolvimento prático do trabalho proposto. Por fim temos a seção 4 utilizada para conclusões do autor, seguida da última seção, número 5, a qual apresenta as referências bibliográficas utilizadas.

2. FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão expostos algumas informações relacionadas à área de IA que podem ser relevantes para o desenvolvimento do presente trabalho.

2.1 INTELIGÊNCIA ARTIFICIAL

Inteligência artificial é uma grande área da computação que ganhou reconhecimento como disciplina há mais de 50 anos, (Poole, 2010).

Um notável precursor da IA e também tido como pai da computação foi Alan Turing. Em 1950 Turing propôs um teste de inteligência, o qual ficou conhecido como **O Teste de Turing.**

O Teste de Turing é basicamente um jogo de imitação entre um interrogador e uma testemunha, na qual o interrogador pode fazer qualquer pergunta que quiser para a testemunha através de uma interface de texto. O computador passa no teste caso o interrogador não consiga dizer se ele está se comunicando com um humano ou uma máquina.

2.1.1 DEFINIÇÕES

Existem diversas definições do que a IA estuda presentes na literatura atual, em (Russel e Norvig, 1995) são discutidas oito diferentes definições provenientes de oito diferentes autores, as quais são divididas em quatro categorias:

* Sistemas que pensam como humanos;
* Sistemas que agem como humanos;
* Sistemas que pensam racionalmente;
* Sistemas que agem racionalmente.

Uma definição objetiva do que é IA está presente em Poole (2010, p.3), a qual diz que, “Inteligência Artificial é o campo que estuda a síntese e análise de agentes computacionais que agem de maneira inteligente”.

Para melhor compreender o que Poole escreveu vamos explicar mais detalhadamente o que é um agente, termo essencial citado na definição acima.

Segundo Russel e Norvig, (1995, p.31), agente é qualquer coisa que pode perceber seu ambiente por meio de sensores e agir sobre este ambiente por meio de atuadores.

Podemos citar como exemplo de agentes o ser humano, robôs, software de computador, etc.

Em um agente robô os atuadores podem ser garras, câmeras, sensores, etc. Já em um humano podemos citar os 5 sentidos, braços, boca.

Segundo Poole (2010, p.3), os agentes atuam de forma inteligente quando executam tarefas adequadas, baseados na situação em que se encontram e levando em consideração seus objetivos; são flexíveis a mudanças de ambiente e também de objetivos; aprendem a partir do meio em que estão imersos; fazem escolhas apropriadas dado que um agente tem memória finita e possui tempo limitado para agir.

2.1.2 SUBÁREAS DA IA

Por ser uma área muito grande dentro da computação a IA é dividida em diversas subáreas. A seguir iremos comentar de forma geral algumas dessas subáreas, procurando enfatizar as que são mais relevantes para o entendimento desse trabalho.

2.1.2.1 REPRESENTAÇÃO DE CONHECIMENTO

No mundo real podemos descrever um problema de maneira mais fácil utilizando linguagem natural. Um exemplo seria dizer que: “é necessário consertar o que estiver quebrado no sistema elétrico da casa”, tal tarefa de descrever o problema é um pouco mais complexa quando precisamos descrevê-lo de forma que um computador consiga entender.

Segundo Poole (2012), para resolver um problema o designer do sistema deve cumprir as seguintes regras:

* Detalhar a tarefa e determinar o que constitui uma solução;
* Representar o problema em uma linguagem na qual um computador pode raciocinar;
* Utilizar um computador para computar uma saída, a qual pode ser uma resposta a ser mostrada para o usuário ou uma ação a ser tomada em determinado ambiente;
* Interpretar uma saída como uma solução para o problema.

O **conhecimento** na IA é a informação sobre um domínio que pode ser usada para resolver problemas naquele domínio. Esse conhecimento precisa ser representado de alguma forma para que possa ser usado pelo computador. Um **esquema de representação** é a forma de conhecimento utilizada por um agente. Uma **base de conhecimento** é a representação de todo o conhecimento que é armazenado por um agente.

2.1.2.2 PROCESSAMENTO DE LINGUAGEM NATURAL (PLN)

Processamento de Linguagem Natural é um domínio de grande importância na IA e especialmente importante para este trabalho onde a área estudada está diretamente ligada ao PLN.

A linguagem natural (ou humana), ao contrário das linguagens de programação, é bastante ambígua e permite que uma frase ou palavra possua diversos significados. Analisando a frase “Ele viu o banco”, é impossível saber se a mesma está se referindo a uma instituição financeira ou a mobília de uma casa.

De uma forma geral, o PLN visa estudar a interação entre o computador e a linguagem humana (linguagem natural).

Segundo Poole (2012) todos os problemas da IA nascem no contexto de tentar resolver “o problema da linguagem natural” e também é dito que isso é tão difícil quanto resolver “o problema da IA” pois qualquer domínio pode ser expresso em linguagem natural.

Abaixo serão citados com base no livro de Poole (2012) algumas das principais razões para se estudar PLN, segundo o autor:

* É desejável que um computador se comunique com um humano de forma que o usuário não necessite aprender uma nova linguagem para utilizar o computador. Fato importante especialmente para usuários comuns;
* Existe uma quantidade muito grande de informação escrita em linguagem natural, ou seja, na forma de livros, revistas, etc. Um sistema que necessite acessar essa informação caso ela esteja disponível online deverá estar apto a processar linguagem natural.
* Muitos dos problemas de IA surgem em uma forma muito clara e explícita no processamento de linguagem natural e, por isso, esse é um bom domínio a ser estudado.

O PLN é base para a recuperação de informação e por isso esse é um tópico que está diretamente ligado aos objetivos principais desse trabalho.

2.1.2.3 APRENDIZAGEM DE MÁQUINA

Aprendizagem de máquina é uma subárea da computação estudada mais especificamente no ramo da Inteligência Artificial. Essa subárea é focada no desenvolvimento e estudo de programas de computador que possuem a capacidade de aprender padrões fornecidos como forma de treinamento e utilizá-los na classificação de novas instâncias seguindo as características das instâncias de teste.

Esse método pode ser aplicado em uma gama muito grande de aplicações, por exemplo, mineração de dados, processamento de linguagem natural, reconhecimento facial, etc.

Um exemplo de aprendizagem de máquina supervisionado pode ser citado como um programa que faz reconhecimento facial. Para que o programa consiga distinguir o que é um rosto e o que não é, são fornecidas diversas instâncias positivas e negativas, para que o computador consiga criar um padrão e posteriormente inferir se uma dada imagem possui um rosto ou não.

2.1.2.4 ALGORITMOS

Veremos a seguir as principais características de alguns algoritmos que são utilizados em Inteligência Artificial para resolver uma grande variedade de problemas.

Podemos dividir os algoritmos a serem descritos em duas classes, sendo elas a classe dos algoritmos utilizados em busca não informada e algoritmos de busca informada.

Dentro da busca não-informada serão descritos os seguintes algoritmos:

* Algoritmo de Busca em Profundidade ou Depth-First-Search (DFS);
* Algoritmo de Busca em Largura ou Breadth-First-Search (BFS);
* Algoritmo de Busca pelo Custo Uniforme.

Os métodos utilizados na busca informada a serem descritos são:

* Melhor-Primeiro (Greedy Search);
* Algoritmo A\*.

Busca não-informada ou busca cega é o nome dado a estratégia de busca que não utiliza informações do domínio do problema para chegar à solução, ela apenas percorre o espaço de estados do problema e verifica se o estado atual é o objetivo ou não.

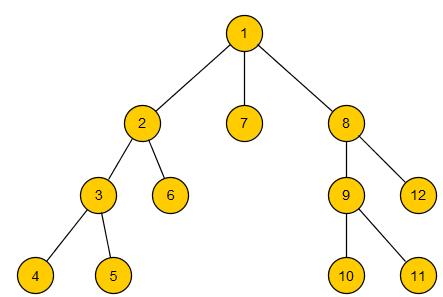
* **Algoritmo de Busca em Profundidade (DFS):** Em uma árvore com raiz *s* o algoritmo visita o nó *v*, adjacente a *s*, e em seguida realiza o mesmo procedimento para os nós subsequêntes à *v*, fazendo retrocesso quando um nó folha é encontrado. Para árvores com fator de ramificação *b* e *m* sendo a profundidade máxima da árvore, a complexidade de tempo desse algoritmo é *O()* e a complexidade de espaço é *O(bm)*. Esse algoritmo não é indicado para problemas que possuam uma árvore com profundidade máxima muito grande ou infinita pois caso o algoritmo comece a expandir um ramo muito profundo ele pode acabar em loop infinito e nunca encontrar uma solução. Dito isso, esse algoritmo não é completo nem ótimo. A figura abaixo ilustra a execução do algoritmo DFS num grafo onde os números representam a ordem com a qual os nós foram visitados:   
  

Figura 1: Ordem com a qual o algoritmo DFS percorre o grafo.

* **Algoritmo de Busca em Largura (BFS):** O BFS, ao contrário do DFS, procura visitar os nós mais próximos à raíz primeiro. O algoritmo começa da raiz e visita todos os nós do nível *d-1* antes de visitar os nós do nível *d*. Segundo Russel e Norvig esse algoritmo tem complexidade de tempo de e complexidade de espaço de onde *b* representa o fator de ramificação da árvore e *d* representa a profundidade da solução. Abaixo temos uma figura ilustrando a ordem com que são percorridos os nós de uma árvore quando aplicamos o aalgoritmo BFS:

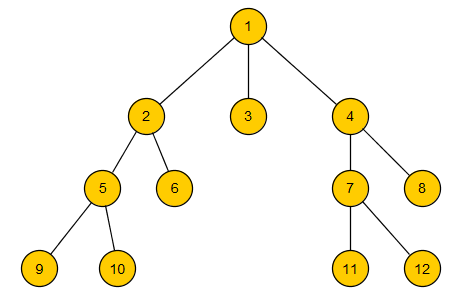
**

Figura 2: Ordem com a qual o algoritmo BFS percorre o grafo.

* **Busca pelo Custo Uniforme:** Esse tipo de busca é utilizado quando há um custo associado aos nós a serem expandidos. O algoritmo de busca pelo custo uniforme utiliza uma função g(n) para verificar o custo da raiz até o próximo nó a ser expandido, assim sendo, ele expande os nós com menor custo primeiro. Caso todos os nós tenham o mesmo custo o algoritmo torna-se idêntico ao BFS. Sabendo que d representa a profundidade da solução e b representa o fator de ramificação, a Busca pelo Custo Uniforme possui complexidade de espaço de O( e complexidade de tempo de O(). Esse método de busca é completo e ótimo caso os custos dos arcos sejam sempre positivos. Abaixo temos uma figura ilustrando a ordem com que os nós seriam expandidos por esse algoritmo:

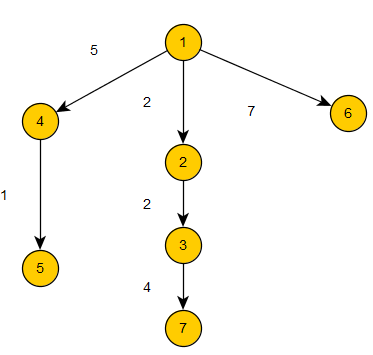


Figura 3: Ordem de visitação dos nós utilizando o algoritmo de Busca pelo Custo Uniforme.

Veremos agora um pouco sobre os algoritmos de Busca-Informada. Esse tipo de busca utiliza informações do domínio do problema para definir qual será a próxima ação a ser tomada, normalmente existe uma função de avaliação que retorna um valor informando o quão próximo determinado nó está da solução, ajudando assim na tomada de decisão e na descoberta de soluções de maneira mais eficiente.

* **Melhor Primeiro (Best-First Search - Greedy):** Esse método de busca tem esse nome pois sempre irá escolher o caminho ótimo local, mesmo que essa escolha não leve a uma solução ótima global. Essa escolha é feita de acordo com alguma função heurística *h* que estima o custo dos nós que estão na fronteira até a solução. Esse algoritmo trata a fronteira como uma fila de prioridades ordenada por *h(n)* e sempre escolhe o nó com *h(n)* mínimo. Esse método não é ótimo pois escolher sempre o mínimo local em determinados problemas pode não ser a melhor escolha e tem complexidade de tempo e de espaço igual a no pior caso, com *m* sendo a profundidade máxima do espaço de busca.
* **Algoritmo A\*:** O algoritmo A\* também utiliza a ideia do Melhor-Primeiro, porém nesse caso o algoritmo sempre irá achar a melhor solução. O A\* utiliza uma função que estima o custo até o nó objetivo, tal função é dada por *f(n) = g(n) + h(n)*. Onde *g(n)* é o custo para sair da raiz e chegar até o nó *n* e *h(n)* é uma heurística que estima o custo de se chegar do nó *n* até algum nó objetivo. Dizemos que para esse algoritmo ser ótimo a função heurística utilizada deve ser admissível, isto é, ela não deve superestimar o custo de um nó até o objetivo. Podemos citar como exemplo prático de heurística admissível num problema de rotas, a distância em linha reta entre dois pontos no mapa, visto que esse sempre será o menor caminho possível.

2.2 ONTOLOGIA

Nesta seção veremos a definição de ontologia em seus diversos meios de aplicação e em seguida será feito uma análise de alguns trabalhos correlatos.

Ontologia em seu conceito filosófico tem a seguinte definição, segundo (Blackburn & Marcondes, 1997), “[...] o ramo da metafísica que diz respeito àquilo que existe”

No ramo de Tecnologia da Informação pode-se dizer que tal palavra é emprestada da filosofia e tem, segundo Gruber (1993) o significado de “uma especificação explícita de uma conceituação do domínio”. Em termos menos formais podemos definir uma ontologia como sendo uma estrutura que define conceitos e a relação entre eles por meio de classes, subclasses e relações. De forma análoga à orientação objeto utilizada em linguagens de programação, cada subclasse em uma ontologia herda as características de sua superclasse e devido a essa propriedade é possível que inferências sejam feitas de forma sistemática, por exemplo, se um *código de cidade* está associado com um *código de estado* e um *endereço* usa esse *código de cidade*, então pode-se concluir que esse *endereço* está assocido ao respectivo *código de estado*.

Diversas formas de representação de conhecimento vêm sendo estudadas, uma delas é a representação de conhecimento por meio de ontologias, esse método permite que conhecimento sobre um dado domínio seja representado de forma estruturada fazendo ligações semânticas entre assuntos relacionados, e permitindo assim, que tais estruturas possam ser utilizadas de diversas formas. Dentre diversas outras aplicações, a recuperação de informação é a área onde a utilização de ontologias é mais comum por permitir uma abordagem semântica na recuperação de informação.

2.2.1 FERRAMENTAS DISPONÍVEIS

Nessa seção iremos citar e descrever brevemente algumas das principais ferramentas disponíveis para a criação e edição de ontologias. Todas as ferramentas descritas abaixo são citadas no artigo *Ontology Editors* da *W3C wiki,* disponível em *www.w3.org/wiki/Ontology\_editors.*

**Protégé:** É uma ferramenta *open-source e* gratuita que permite a edição e criação de ontologias. Foi desenvolvida pelo *Stanford Center for Biomedical Informatics Research (BMIR)*, na Universidade de Medicina de Stanford.

Possui mais de 200,000 usuários registrados e é a ferramenta mais utilizada no mundopara a criação e edição de ontologias.

O Protégé possui uma versão *Web* dotada de interface customizável facilitando assim o uso para usuários novatos no ramo de criação de ontologias*.* Além disso pelo fato da ferramenta ser *Web* ela permite de maneira fácil a criação, *upload,* edição e compartilhamento de ontologias de forma colaborativa com outros usuários permitindo edição e visualização em tempo real.

A Protégé possui também uma versão *Desktop* que permite a criação e edição de uma ou mais ontologias em um só *workspace* por meio de uma interface completamente customizável. Operações de refatoramento são disponíveis incluindo a junção de duas ou mais ontologias.

Existe também de tempos em tempos um treinamento oferecido na Universidade de Stanford, CA, com duração de três dias voltado para usuários iniciantes e intermediários da ferramenta.

**NeOn Toolkit:** É um editor de ontologias *open-source* multi-plataforma voltado principalmente para projetos de grande porte.

Segundo o próprio *site, http://neon-toolkit.org,* o principal objetivo dessa ferramenta é facilitar o manuseio de multiplas ontologias que já existem em um determinado contexto e são desenvolvidas colaborativamente e por isso estão constantemente sofrendo alterações e evoluindo.

**Neologism:** É uma plataforma *open-source* de publicação de vocabulário para a *Web of Data.* Foi desenvolvida em PHP e atualmente ainda está na versão beta.

Nela é possível utilizar o padrão RDF, muito comum na criação de ontologias, a ferramenta também dá suporte a uma parte da OWL *(Web Ontology Language).*

**TopBraid Composer:** O TopBraid Composer oferece suporte para a construção, edição e teste de ontologias e grafos RDF. É implementado como um plugin para o *Eclipse*

Dentre suas principais funcionalidades podemos destacar:

* Possui suporte visual para edição de grafos RDF e diagramas de classe;
* Conversão automatizada de planilhas nos formatos Excel, UML e outros;
* Suporte ao protocolo SPARQL, o qual é uma linguagem de consulta específica para o formato RDF;
* Totalmente compatível com os padrões impostos pela W3C.

A ferramenta está disponível no site *www.topquadrant.com/tools/modeling-topbraid-composer-standart-edition/* em duas versões, a versão *Standart* a qual é gratuita e permite o uso do software com funcionalidades limitadas e a versão *Maestro* a qual possui todas as funcionalidades da versão *Standart* complementada de outras funcionalidades como o suporte para o desenvolvimento de aplicações utilizando ferramentas como *SPIN* e *SPARQLMotion*.

2.2.2 FORMAS DE MODELAGEM DE UMA ONTOLOGIA

Na década de 1990 houve um crescente aumento no interesse das pessoas em se construir e utilizar ontologias. Nessa epoca o desenvolvimento de ontologias era tido como uma ‘arte’ ao invés de uma atividade de engenharia dotada de uma metodologia formal que guiasse sua construção.

Ao final da década de 1990, mais precisamente em 1996 aconteceu o primeiro *workshop* sobre metodologias para se construir ontologias, a partir desse ano houve um aumento crescente na disseminação e definição de metodologias para a criação de ontologias**.**

Abaixo serão citadas algumas das principais metodologias para a criação de ontologias.

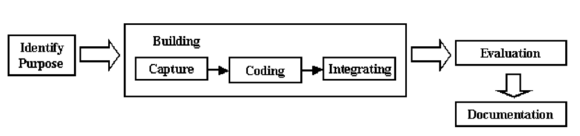
* **Método Uschold e King (1995):** Esse foi o primeiro método a ser proposto para a construção de ontologias. Segundo o método para se construir uma ontologia deve-se seguir quatro passos principais: (1) identificar o propósito da ontologia, (2) construí-la, (3) avaliá-la e por fim (4) documentá-la. Na figura abaixo podemos ver os principais processos contidos no método de Uschold e King.

Figura 4: Principais processos do método de Uschold e King.

A seguir iremos explicar de forma breve os 4 passos principais mostrados na figura acima.

1. **Identificar o Objetivo (Identify Purpose):** Deve-se deixar claro o porque de se estar construindo determinada ontologia e qual será o seu uso depois que estiver pronta.
2. **Construindo (Building):** Essa fase é dividida em três passos
3. **Captura (Capture):** Identificar os conceitos chave e seus relacionamentos. Identificação dos termos que serão referidos aos conceitos e suas relações. Produção de definições não ambiguas e precisas para os conceitos e relações.
4. **Codificação (Coding):** Representar o conhecimento adquirido em *i* utilizando uma linguagem formal.
5. **Integração (Integrating):** Deve-se definir se serão utilizadas ontologias que já existem na ontologia a ser criada. Isso deve ocorrer durante os passos de *Captura* e *Codificação.*
6. **Avaliação (Evaluation):** Nessa fase deve-se fazer a avaliação e verificar se a ontologia criada atende às expectativas.
7. **Documentação (Documentation):** Nessa fase deve ser feita a documentação da ontologia.

Além do método descrito acima existem diversos outros métodos como o *Methontology, On-To-Knowledge,* etc.

2.3 TRABALHOS CORRELATOS

Nesta seção iremos expor alguns trabalhos já realizados na área de ontologia ligados à recuperação de informação, serão apresentados os principais objetivos de cada trabalho, suas características mais relevantes e os resultados obtidos de forma sintetizada.

No artigo de Paz-Trillo et al. (2005), foi produzido um programa dotado de interface gráfica com o objetivo de auxiliar na recuperação de pequenos trechos de vídeos dentro de uma coleção específica fornecida pelo usuário com a ajuda do uso de ontologias específicas para determinado domínio. A ontologia criada nesse trabalho foi produziada utilizando a ferramenta específica para criação de ontologias *protégé*. Dentro do programa é possível escrever *queries* que representem o que o usuário deseja buscar no vídeo e o programa então se encarrega de procurar a seção do vídeo com a qual a *query* informada pelo usuário melhor se encaixa. Isso é feito com o auxilio de uma ontologia específica para a área. Os resultados obtidos segundo os autores melhoraram a precisão e o recall das *queries* em comparação com sistemas tradicionais. Além disso, os autores mostraram também que o uso de uma ontologia mais rica, com propriedades específicas para cada classe, obtiveram resultados melhores, indicando que quanto mais rica for a ontologia, melhor serão os resultados obtidos na recuperação de informação.

O relatório de pesquisa apresentado por Edberto Ferneda (2013) apresenta um modelo de recuperação de informação baseado no Modelo Espaço Vetorial, que utiliza ontologias como elemento normalizador das representações, tanto de documentos quanto das necessidades de informação do usuário. Um protótipo de sistema de recuperação de informação chamado *OntoSmart* foi desenvolvido pelo autor utilizando os conceitos previamente citados, porém no relatório de pesquisa não foram apresentadas análises de resultados para que fosse possível chegar a uma conclusão com relação à eficácia e eficiência do método proposto.

O artigo de Fernández (2011) também aborda a utilização de ontologias na recuperação de informação. A maior contribuição do artigo é um modelo inovador e compreensivo de busca semântica. O artigo busca integrar os prós da busca por palavra chave junto à busca semântica como forma de otimizar a recuperação de informação. É concluído pelos autores que abordagens de recuperação de informação utilizando semântica podem ser auxiliadas pela *Semantic Web* e utilizadas em conjunto com métodos de busca por palavra chave visando melhores resultados. Além disso os autores apontam que o tamanho e a heterogeneidade da *Web*, a escassez de conhecimento semântico, dentre outros fatores, são os principais responsáveis pela demora na aplicação do paradigma de recuperação semântica na *Web.*

De acordo com Kara (2012), sistemas de recuperação de informação semântica podem ser altamente escaláveis utilizando a abordagem de representar o mundo todo como pequenos modelos independentes. No trabalho foi feito um sistema de extração e recuperação de informação baseado em ontologias aplicado ao domínio do futebol. O sistema foi implementado utilizando tecnologias relacionadas à *Semantic Web* e avaliado em comparação com sistemas tradicionais. Foi concluído que o sistema apresentado é nitidamente melhor que os métodos tradicionais e também se sobressai em comparação a métodos de expansão de *query*.

Existe uma quantidade muito grande de documentos que estudam a criação e utilização de ontologias, em específico aplicando os conceitos junto à recuperação de informação. O trabalho de conclusão de curso aqui proposto possui características muito parecidas principalmente com os artigos de Kara (2012) e Fernández (2011). A diferença principal entre o trabalho proposto e os artigos citados acima é o domínio de aplicação da recuperação de informação, que é a área de Inteligência Artificial no trabalho proposto.

3. DESENVOLVIMENTO

Com base no levantamento teórico realizado em relação a ontologias e alguns dos conceitos teóricos de Inteligência Artificial, foi desenvolvido o protótipo de busca com base em ontologia, o qual será descrito na sequência.

3.1 ESTUDO DE FERRAMENTAS ESPECÍFICAS PARA O DESENVOLVIMENTO DE ONTOLOGIAS

Para o desenvolvimento de uma ontologia podem ser utilizados diversos tipos de ferramentas, podendo até mesmo se fazer o uso técnicas de diagramação. Contudo, de modo a facilitar essa atividade foram estudados diferentes softwares que permitem a criação, edição e manipulação de ontologias como o *Protégé, NeOn toolkit, OntoBuilder, Web Protégé, TopBraid Composer,* dentre outras.

Partindo do estudo feito sobre tais ferramentas foi possível tomar a decisão de qual das ferramentas utilizar para a modelagem da ontologia proposta no trabalho.

A ferramenta escolhida foi o *Protégé* em sua versão *Desktop*, isso ocorreu devido a diversos fatores como a facilidade de utilização, vasta quantidade de tutoriais e materiais descrevendo e ensinando como utilizar a ferramenta.

3.2 CRIAÇÃO DA ONTOLOGIA

O desenvolvimento de uma ontologia é uma tarefa que requer tempo e muito estudo sobre o domínio a ser modelado e ferramentas a serem utilizadas.

A Ontologia desenvolvida nesse trabalho começou a ser rascunhada utilizando uma planilha *Excel,* para que pudesse começar a tomar forma antes de ser atrelada a um softwareespecífico para esse propósito*.*

Na planilha *Excel* foram adicionados os principais conceitos de Inteligência Artificial utilizando como base os livros de *Russel & Norvig* e o sistema de classificação da *ACM*, chamado de *Computing Classification System* (CCS).

A modelagem em *Excel* seguiu um modelo de cascata para relacionar as principais áreas e subáreas da Inteligência Artificial. Na página seguinte podemos ver como ficou o resultado de uma parte da Ontologia desenvolvida no *Excel*.

A Ontologia consiste de uma tabela com cinco colunas onde a última delas mostra a fonte utilizada para classificar alguns termos e as outras quatro representam as classes e subclasses da ontologia sendo que as classes localizadas mais à esquerda são mais abrangentes do que as mais à direita.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Processamento de Linguagem Natural | Recuperação de Informação | Modelo de Recuperação Booleana |  | Russel & Norvig |
|  |  | Rankeamento e Atribuição de Peso |  | ACM |
|  |  | Recuperação XML |  |  |
|  |  | Extração de Relações Entre Termos |  | Russel & Norvig |
|  |  | Classificação de Documentos | Teorema de Bayes | Russel & Norvig |
|  |  | Extração de Dados |  |  |
|  | Geração de Linguagem Natural |  |  |  |
|  | Reconhecimento de Fala |  |  |  |
|  | Tradução de Linguagem |  |  |  |
|  | Optical Character Recognition | Pré-Processamento | Binarização |  |
|  |  |  | Detecção de linhas |  |
|  |  | Pós-Processamento |  |  |

*Quadro 3: Resultado de uma parte da ontologia modelada visualizada no Excel.*

Com a planilha finalizada a ontologia foi modelada utilizando o software *Protégé* versão 5.0.0 para *desktop,* a qual pode ser encontrada gratuitamente no endereço [*http://protege.stanford.edu/products.php*](http://protege.stanford.edu/products.php)*.*

Para que a ontologia seja criada basta selecionar a entidade *Thing* dentro da aba *Entities* e adicionar as subclasses desejadas. Além de permitir a criação de classes e subclasses o *Protégé* possibilita a utilização de diversos outros recursos não explorados nesse trabalho, como por exemplo, o uso de relações entre duas classes, a descrição de tipo da classe, etc.

Após terminar a modelagem da ontologia é necessário que ela seja salva no formato correto para que o *script* leitor de arquivos *.owl,* descrito na seção 3.3, possa processar os dados de forma correta.

O *Protégé* permite que a ontologia seja salva em diversos formatos, dentre eles o *RDF/XML, Turtle, OWL/XML*. O formato que desejamos é o *RDF/XML* e para isso devemos clicar em *File, Save as...* e selecionar a opção *RDF/XML Syntax.*

Ao salvar a ontologia nesse formato um arquivo *.owl* será criado, o qual servirá de entrada para o *script* leitor de ontologias.

Podemos ver na figura abaixo a interface do software utilizado na criação da ontologia (*Protégé)* e algumas das classes da ontologia criada.

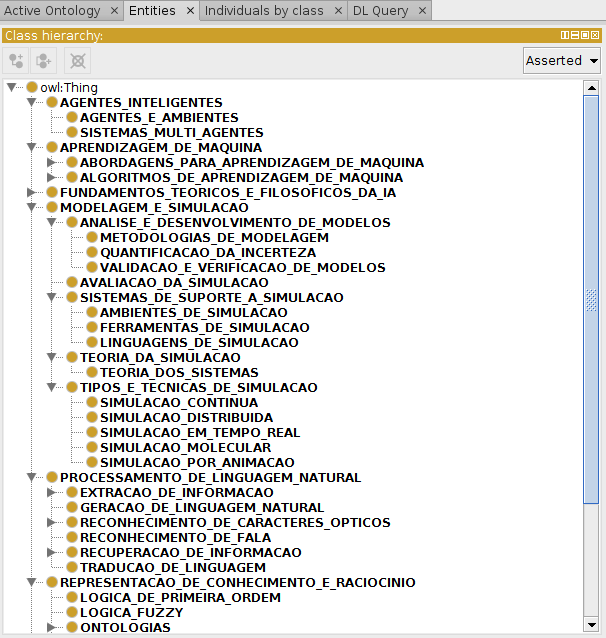


Figura 5: Interface do software Protégé mostrando classes e subclasses da ontologia.

3.3 IMPLEMENTAÇÃO DE UM SCRIPT PARA PROCESSAR O ARQUIVO .OWL

A linguagem escolhida para o desenvolvimento desse trabalho, o *Python*, possui algumas bibliotecas que facilitam a leitura de arquivos *.owl* como a *owlready, rdflib* e a *ontospy*.

A biblioteca escolhida para o processamento da ontologia foi a *ontospy*, pois pareceu ser a única a fornecer um *output* padronizado que representasse a ontologia modelada.

A biblioteca *ontospy* funciona apenas em sistemas *Linux*, abaixo veremos uma figura com um pequeno trecho de código utilizado para criar um objeto *Python* a partir da ontologia. A partir dele é possível executar diversas operações. No trecho de código abaixo o nome de todas as classes do arquivo .*owl* são printadas na tela porém não são mostrados na figura.

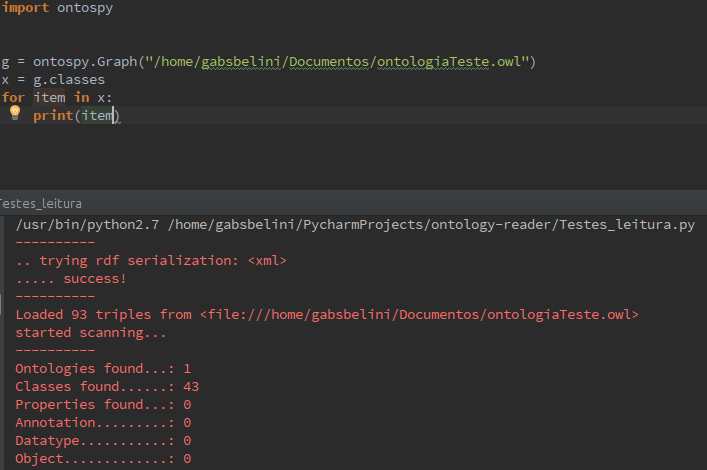


Figura 6: Trecho de código em Python, mostrando utilização da biblioteca ontospy.

A figura acima mostra um trecho de código exemplificando o uso da biblioteca ontospy para printar todas as classes da ontologia na tela utilizando Python 2.7.

O *output* desejado para o programa principal é obtido através da função *printClassTree(),* essa função imprime todas as classes da ontologia na saída *stderr* do computador numa forma esquemática. O *output* gerado utiliza quatro hífens para identar cada nível da ontologia, a seguir temos uma imagem exemplificando este *output.*

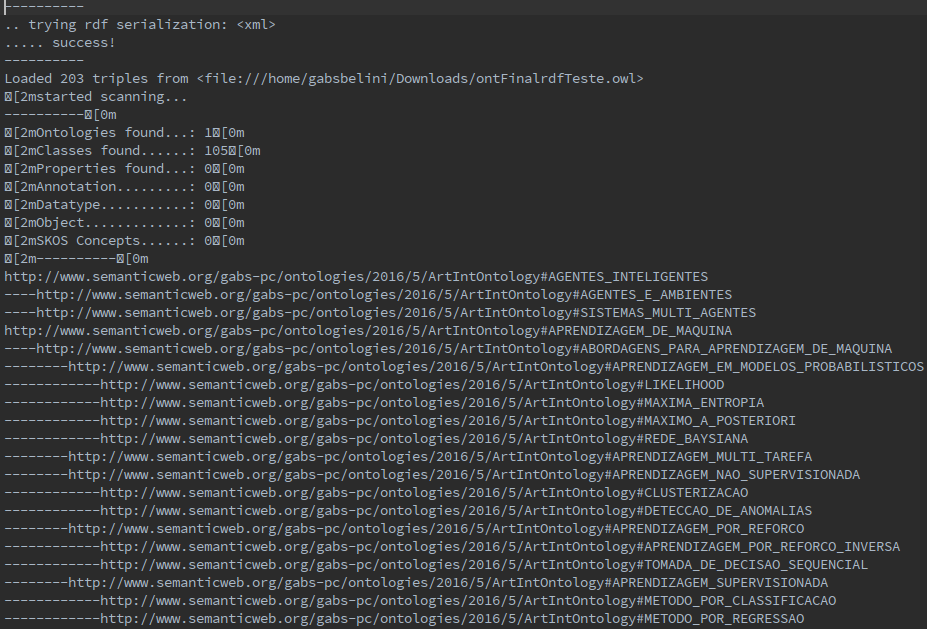


Figura 7: Resultado da função printClassTree() da biblioteca ontospy.

Esse *output* deve ser salvo em um arquivo *.txt* pois será utilizado no programa principal. Isso pode ser feito com o seguinte comando: *python leitorOntologia.py 2> saida.txt*.

**3.4 IMPLEMENTAÇÃO DO PROTÓTIPO DE BUSCAS AUTOMATIZADAS COM INTERFACE GRÁFICA**

Novamente utilizando *Python 2.7*, foi implementado um programa com interface gráfica para que o usuário final possa efetuar buscas no *google* utilizando uma ontologia.

A figura abaixo representa o diagrama de caso de uso para o protótipo

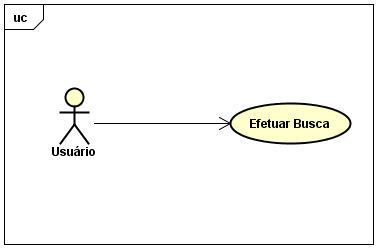


Figura 8: Caso de uso do protótipo desenvolvido.

Para que o usuário possa executar o protótipo e posteriormente efetue a busca, algumas configurações são necessárias. Abaixo serão citadas em tópicos as configurações necessárias para que o protótipo possa ser executado.

* *Python 2.7* Instalado;
* Biblioteca de testes automatizados *Selenium* Instalada*;*
* Download do ChromeDriver;
* Alteração da linha driver = webdriver.Chrome('/home/gabsbelini/chromedriver') no protótipo, *prototipo\_final.py,* para que o caminho seja o mesmo de onde está localizado o ChromeDriver em seu computador. Vale ressaltar que em sistemas Linux o valor acima é um valor válido porém em sistemas windows é necessário adicionar a extensão do arquivo *.exe* para que o mesmo funcione corretamente, assim sendo, um exemplo em windows ficaria da seguinte forma: driver = webdriver.Chrome(' C:\Users\gabs\Downloads\chromedriver.exe);

Após os itens acima terem sido satisfeitos basta rodar o protótipo com o seguinte comando no terminal:

* *python prototipo\_final.py*

A interface gráfica será parecida com a da figura a seguir:



Figura 9: Imagem ilustrando a interface principal do protótipo.

Com a interface aberta basta selecionar os termos que deseja pesquisar clicando em seus respectivos *checkboxes* e após isso clicar em *Buscar*. Abaixo podemos ver um exemplo da interface com algumas classes selecionadas

O Google Chrome será aberto automaticamente e de forma também automática uma busca com todos os termos selecionados previamente, separados por espaço, será realizada.

Segue abaixo um exemplo do resultado de uma busca feita pelo protótipo:

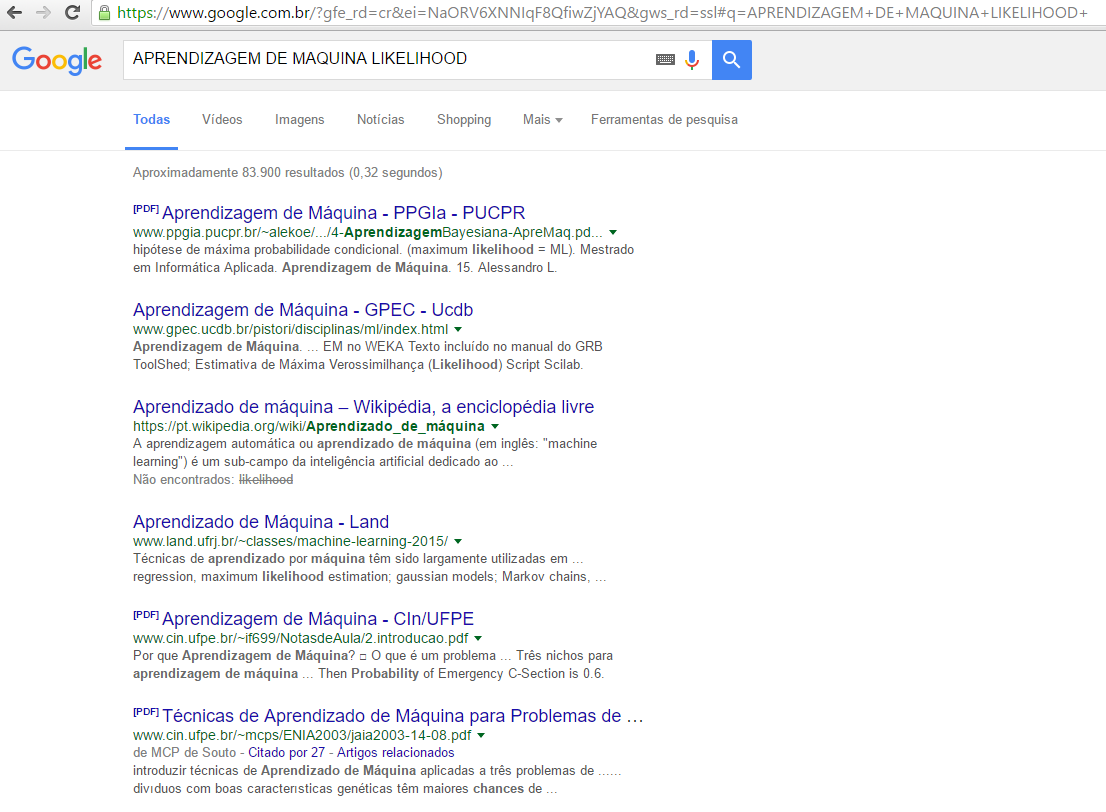


Figura 10: Resultado apresentado pelo protótipo após efetuar a busca.

A imagem acima mostra o resultado apresentado pelo programa após selecionar os checkboxes “Aprendizagem de Máquina” e “Likelihood” na interface e clicar no botão buscar.

Vale ressaltar que, apesar do exemplo apresentado no protótipo ter sido feito com uma ontologia da área de Inteligência Artificial, o protótipo desenvolvido é genérico o suficiente para aceitar qualquer tipo de ontologia independente da profundidade e do número de classes que ela possuir. Isso desde que o arquivo contendo os dados da ontologia usado como input no protótipo seja um arquivo *.txt* do output gerado pela função *printClassTree()* .

**3.4.5 INFORMAÇÕES SOBRE O CÓDIGO IMPLEMENTADO**

Abaixo será discutido um pouco sobre as principais bibliotecas utilizadas no desenvolvimento do programa e também serão apresentados alguns trechos de código.

A principal função utilizada no protótipo é a responsável pelo posicionamento e criação das *checkboxes* dentro da janela principal do programa.

Segue abaixo um trecho de código comentado onde é possível verificar como são criadas as variáveis para cada *checkbox,* a criação do objeto *ck* instanciado a cada vez que a função é chamada e por fim seu posicionamento na janela utilizando o padrão *grid layout*.

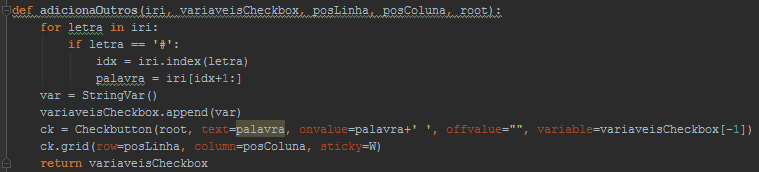


Figura 11: Trecho de código em Python, mostrando a função adicionaOutros()

O código acima contém a função responsável por tratar os casos gerais de posicionamento de classes na interface. Essa função extrai do iri o nome a ser utilizado na interface, cria uma variável para a checkbox, adiciona essa variável a uma lista, cria o objeto do tipo checkbox *“ck”*, posiciona esse objeto corretamente utilizando grid layout e por fim retorna uma lista contendo todas as variáveis ligadas aos checkboxes criados.

A seguir temos o trecho de código que ilustra a biblioteca responsável por abrir o navegador definido e efetuar a busca no google de forma automática. Essa biblioteca é chamada *Selenium* e é uma biblioteca utilizada principalmente em testes automatizados de sistemas.

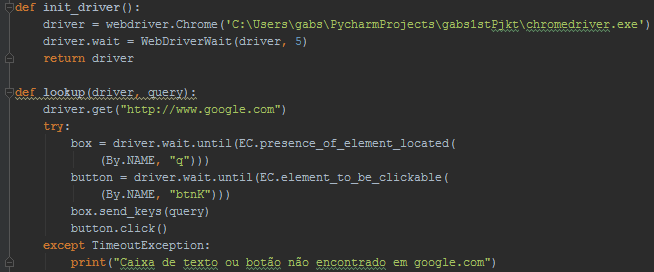


Figura 12: Trecho de código em Python mostrando as funções da biblioteca Selenium utilizadas.

As duas funções acima são responsáveis por inicializar o chrome e efetuar a busca da *“query”* escolhida pelo usuário através da interface.

No início do desenvolvimento a abordagem escolhida havia sido a de utilizar a API do *google* para criar uma URL dentro do código em *Python* com os parâmetros desejados, efetuar uma requisição via *urllib* para esse endereço e extrair o código *HTML* dessa página.

Isso não foi possível pois recentemente o *google* fez algumas mudanças na forma com que permite que usuários efetuem requisições *HTTP* ao seu mecanismo de busca, decorrente disso a primeira abordagem tornou-se mais complicada de ser colocada em prática e a solução foi utilizar a biblioteca de testes citada acima, o *Selenium*.

O *Selenium* é normalmente utilizado para realizar testes de interface como se fosse um robô, preenchendo os campos de informação definidos pelo programador e por fim testando o sistema.

Apesar de ser uma biblioteca de testes, o *Selenium* não foi utilizado com esse propósito no protótipo construído.

**4. CONCLUSÕES**

A utilização e criação de ontologias é de grande importância para a sociedade, sua utilização abrange diversos domínios de aplicação, dentre eles desde a classificação taxonômica de palavras utilizadas em sites de busca, como o *Yahoo,* até a categorização de produtos e suas funcionalidades como na *Amazon.com.*

O objetivo geral desse trabalho era o de construir uma ontologia para a área de Inteligência Artificial, procurando categorizar todos os principais termos dentro desse domínio específico. Para desenvolver tal tarefa fez-se o uso da ferramenta *Protégé,* específica para esse propósito. Foram criadas classes e subclasses com base em livros como o de *Russel & Norvig* e também estudando o sistema de classificação da ACM, *Computing Classification System* (CCS).

A criação de uma ontologia para um determinado domínio requer que o autor responsável entenda de forma geral os principais conceitos desse domínio e como cada elemento a ser modelado se relaciona com os outros. Para que esse entendimento fosse atingido foi necessário muito estudo sobre os principais conceitos da Inteligência Artificial e como são suas respectivas relações. Uma ontologia sobre um domínio pode ser modelada de diversas formas, porém para que faça sentido é necessário que o autor crie as relações de forma correta.

Ontologias estão ficando cada vez mais comuns de serem encontradas. Com isso o seu campo de aplicação também tem ampliado, para ter uma noção universal das áreas de aplicação e o que vêm sendo feito com o auxilio de ontologias foram estudados diversos artigos que abrangem esse assunto, alguns deles são citados na seção 2.3.

A modelagem de ontologias pode ser feita de diversas formas. É possível criar uma ontologia utilizando apenas uma planilha *Excel* como foi feito no princípio desse trabalho. Porém existem diversos softwares disponíveis que auxiliam nesse desenvolvimento e facilitam esse trabalho. Sabendo disso foram estudados alguns dos principais softwares utilizados e escolhido o *Protégé* para a criação da ontologia proposta.

De posse da ontologia já modelada foi possível desenvolver um protótipo o qual auxilia o usuário a pesquisar na internet de forma simples as classes que ele desejar. A ontologia utilizada no protótipo foi a de Inteligência Artificial, porém caso necessário o usuário pode desenvolver sua própria ontologia e utilizar o protótipo da mesma forma para efetuar buscas sobre outra ontologia.

O protótipo desenvolvido é capaz de realizar todas as tarefas propostas no trabalho, porém pode também ser melhorado em diversos aspectos. Em trabalhos futuros é possível que os dois programas criados (*leitorOntologia.py* e *prototipo\_final.py*) funcionem de forma integrada, melhorias na interface podem ser implementadas, é possível também adicionar conectivos lógicos às checkboxes, permitindo fazer buscas mais específicas, e por fim pode-se também permitir que o usuário faça a seleção de qual *browser* ele deseja utilizar.

5. REFERÊNCIAS BIBLIOGRÁFICAS

RUSSEL, Stuart Jonathan; NORVIG, Peter. **Artificial Intelligence:**A Modern Approach. New Jersey: Prentice-hall do Brasil, 1995. 932 p. Disponível em: <http://www.cin.ufpe.br/~tfl2/artificial-intelligence-modern-approach.9780131038059.25368.pdf>. Acesso em: 13 nov. 2015.

GÓMEZ-PÉREZ, Asunción; FERNÁNDEZ-LÓPEZ, Mariano; CORCHO, Oscar. **Ontological Engineering.**Madrid: Springer, 2004. 401 p.

W3C. Semantic Web. World Wide Web Consortium (W3C). Disponível em: <http://www.w3.org/standards/semanticweb/ontology>. Acesso em: 13 de novembro de 2015.

PAZ-TRILLO, Christian; WASSERMANN, Renata; BRAGA, Paula P.. **An information retrieval application using ontologies.**2005. Disponível em: <http://www.scielo.br/scielo.php?pid=S0104-65002005000300002&script=sci\_arttext>. Acesso em: 31 nov. 2015.

FERNÁNDEZ, Miriam et al. **Semantically enhanced Information Retrieval: An ontology-based approach.**2011. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1570826810000910>. Acesso em: 16 nov. 2015.

LÓPEZ, Fernández. **Overview Of Methodologies For Building Ontologies.**2003. Disponível em: < http://www.cs.upc.edu/~bejar/aia/aia-web/4-fernandez.pdf>. Acesso em: 15 fev. 2016.

KARA, Soner et al. **An ontology-based retrieval system using semantic indexing.**2012. Disponível em: <http://www.sciencedirect.com/science/article/pii/S030643791100113X>. Acesso em: 16 nov. 2015.

FERNEDA, Edberto. **Ontologia como recurso de padronização terminológica em um Sistema de Recuperação de Informação.**2013. Disponível em: <https://www.marilia.unesp.br/Home/Instituicao/Docentes/EdbertoFerneda/pos-doutorado.pdf>. Acesso em: 16 nov. 2015.

MORAIS, Edison Andrade Martins; AMBRÓSIO, Ana Paulo L.. **Ontologias: conceitos, usos, tipos, metodologias, ferramentas e linguagens.**Goiás: Instituto de Informática, 2007. 22 p. Disponível em: <http://www.inf.ufg.br/sites/default/files/uploads/relatorios-tecnicos/RT-INF\_001-07.pdf>. Acesso em: 15 fev. 2016.

ASSOCIATION FOR COMPUTING MACHINERY. **Computing Classification System.** 2016. Disponível em: <http://dl.acm.org/ccs/ccs\_flat.cfm>. Acesso em: 01 jun. 2016.

NOY, Natalya F.; MCGUINNESS, Deborah L.. **Ontology Development 101: A Guide to Creating Your First Ontology.**Disponível em: <http://protege.stanford.edu/publications/ontology\_development/ontology101-noy-mcguinness.html>. Acesso em: 1 jul. 2016.

**APÊNDICE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Processamento de Linguagem Natural | Recuperação de Informação | Modelo de recuperação Booleana |  | Russel & Norvig |
|  |  | Rankeamento e Atribuição de Peso |  | ACM |
|  |  | Recuperação XML |  |  |
|  | Extração de Informação | Extração de Relações Entre Termos |  | Russel & Norvig |
|  |  | Classificação de Documentos | Teorema de Bayes | Russel & Norvig |
|  |  | Extração de Dados |  |  |
|  | Geração de Linguagem Natural |  |  | ACM |
|  | Reconhecimento de Fala |  |  | ACM |
|  | Tradução de Linguagem |  |  | ACM |
|  | Reconhecimento de Caracteres Opticos | Pré-Processamento | Binarização |  |
|  |  |  | Detecção de Linhas |  |
|  |  | Pós-Processamento |  |  |
| Representação de Conhecimento e Raciocínio | Redes Semanticas |  |  | ACM |
|  | Raciocínio Probabilístico |  |  | ACM |
|  | Lógica Fuzzy |  |  | ACM |
|  | Lógica de Primeira Ordem |  |  | Russel & Norvig |
|  | Ontologias | Engenharia de Ontologias | Editores de Ontologias | ACM |
|  |  | Linguagens para Ontologias | OntoUML |  |
|  |  |  | OWL |  |
|  |  |  | IDEF5 |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Resolução de Problemas | | Estratégias de Busca | | Busca Não Informada | | Busca em Profundidade | | Russel & Norvig | |
|  | |  | |  | | Busca em Largura | | Russel & Norvig | |
|  | |  | |  | | Busca pelo Custo Uniforme | | Russel & Norvig | |
|  | |  | |  | | Método Iterativo Incremental | | Russel & Norvig | |
|  | |  | | Busca Informada | | Melhor Primeiro | | Russel & Norvig | |
|  | |  | |  | | A\* | | Russel & Norvig | |
|  | |  | |  | | Subida de Encosta | | Russel & Norvig | |
|  | |  | |  | | Têmpera Simulada | | Russel & Norvig | |
|  | | Problemas de Satisfação de Restrições | |  | |  | | Russel & Norvig | |
| Aprendizagem de Máquina | |  | | Aprendizagem Supervisionada | | Ranking | | ACM | |
|  | |  | |  | | Método por classificação | | ACM | |
|  | |  | |  | | Método por regressão | | ACM | |
|  | |  | | Aprendizagem não Supervisionada | | Clusterização | | ACM | |
|  | |  | | Aprendizagem por Reforço | | Tomada de Decisão Sequencial | | ACM | |
|  | |  | |  | | Aprendizagem por Reforço Inversa | | ACM | |
|  | |  | | Aprendizagem Multi-Tarefa | |  | | ACM | |
|  | | Abordagens para Aprendizagem de Máquina | | Classificação e Árvores de Regressão | |  | | ACM | |
|  | |  | | Aprendizagem em Modelos Probabilísticos | | Likelihood | | ACM | |
|  | |  | |  | | Máxima Entropia | | ACM | |
|  | |  | |  | | Máximo a posteriori | | ACM | |
|  |  | |  | | Rede Baysiana | | ACM | |
|  |  | | Redes Neurais | |  | | ACM | |
|  |  | | Processos de Decisão de Markov | |  | | ACM | |
|  | Algoritmos de Aprendizagem de Máquina | | Algoritmos de Árvore de Decisão | |  | |  | |
| Visão Computacional | Tarefas da Visão Computacional | | Inspeção Visual | |  | | ACM | |
|  |  | | Visão para Robótica | |  | | ACM | |
|  |  | | Sumarização de Vídeo | |  | | ACM | |
|  | Aquisição de Vídeo e Imagem | | Calibração de Câmera | |  | | ACM | |
|  |  | | Fotografia Computacional | |  | | ACM | |
|  |  | | Captura de Movimento | |  | | ACM | |
|  | Formas de Representação da Visão Computacional | | Representação de Imagens | |  | | ACM | |
|  |  | | Representações Hierárquicas | |  | | ACM | |
|  | Problemas da Visão Computacional | | Segmentação de Vídeo | |  | | ACM | |
|  |  | | Segmentação de Imagem | |  | | ACM | |
|  |  | | Detecção de Objetos | |  | | ACM | |
|  |  | | Identificação de Objetos | |  | | ACM | |
| Fundamentos teóricos e filosóficos da Inteligência Artificial | Ciência Cognitiva | |  | |  | | ACM | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Teoria Computacional da Mente |  |  | ACM |
| Modelagem e Simulação | Análise e Desenvolvimento de Modelos | Metodologias de Modelagem |  | ACM |
|  |  | Validação e Verificação de Modelos |  | ACM |
|  |  | Quantificação da Incerteza |  | ACM |
|  | Teoria da Simulação |  |  | ACM |
|  | Tipos e Técnicas de Simulação | Quantificação da Incerteza |  | ACM |
|  |  | Simulação Molecular |  | ACM |
|  |  | Simulação Distribuída |  | ACM |
|  |  | Simulação Contínua |  | ACM |
|  |  | Simulação em Tempo Real |  | ACM |
|  |  | Simulação por Animação |  | ACM |
|  | Sistemas de Suporte à Simulação | Ambientes de Simulação |  | ACM |
|  |  | Linguagens de Simulação |  | ACM |
|  |  | Ferramentas de Simulação |  | ACM |
|  | Avaliação da Simulação |  |  | ACM |
| Agentes Inteligentes | Agentes e Ambientes |  |  | ACM |
|  | Sistemas Multi-Agentes |  |  | ACM |