Relatório de I.A.: OWL, Parte 1

Cauê Baasch de Souza
João Paulo Taylor Ienczak Zanette

4 de Outubro de 2018

TO-DO:

- Adicionar enunciado da 2ª parte (parte prática);
- Escolher domínio dos exemplos.

1 Geral

- Será necessário entregar com a parte 1 e uma breve descrição do domínio escolhido, bem como das principais classes e propriedades definidas, ontologias importadas (quando aplicável), e testes realizados com os indivíduos (o que foi inferido a partir da terminologia);
- Além disso, o arquivo RDF/OWL com o modelo conceitual.

2 Domínio escolhido

2.1 Domínio escolhido

Linguagens, suas implementações e programas descritos nelas.

2.2 Principais Classes:

- **Language:** Representa linguagens de programação (e.g. Python, Ruby, C...);
- **Implementation:** Representa uma implementação de uma linguagem de programação. Por exemplo: CPython, Pypy e MicroPython são implementações de Python;
- **Compiler:** Representa um compilador, podendo incluir JITs. Identifica-se um compilador como AOT (*Ahead of Time*) pela propriedade `aot`. A negativa dessa propriedade identifica o compilador como JIT.
- **Interpreter:** Representa um interpretador.

3 Parte 1: Pesquisa Teórica

Observações:

- O resultado da parte 1 deve ser um pequeno texto explicando o **entendimento** de vocês sobre os tópicos sugeridos. Portanto, o texto deve ter **no máximo** 2 páginas;
- Citar todas as fontes utilizadas para a pesquisa (Wikipédia também serve).

1. Em OWL 2, qual é a diferença entre os axiomas de class `subClassOf` e `equivalentTo`?
   - Apresente as definições de cada um e exemplos de uso dos dois, dentro do domínio escolhido pela dupla para a parte prática.
   - Descreva especialmente a diferença dos axiomas de classe quanto às inferências possíveis, ou seja, teste os exemplos no seu domínio e descreva as inferências.

Solução:

`subClassOf:` Dadas duas classes A e B e individuos que pertençam a apenas uma delas. Ao se fazer `subClassOf(:A :B)`, se está indicando que todo elemento pertencente a A também pertence a B, **mas o contrário não necessariamente**, ou seja: $A \subseteq B$. Sendo assim, para $\{1, 2, 3\} \in A$ e $\{4, 5, 6\} \in B$, `subClassOf(:A :B)` fará com que os indivíduos que pertencem a A e B sejam, respectivamente, $\{1, 2, 3\}$ e $\{1, 2, 3, 4, 5, 6\}$;

Exemplo: No domínio escolhido para a parte prática, foi definido que toda `VirtualMachine` é também um `Program`. Porém, faz sentido que nem todo programa seja uma `VirtualMachine`: nesse caso, não é possível inferir, por exemplo, que o GCC seja uma `VirtualMachine` mesmo que seja um `Program`. Porém, é possível inferir que o GCC é um `Program` simplesmente por defini-lo como um `Compiler`, afinal todo compilador é também um programa.

`equivalentTo:` Dadas duas classes A e B e indivíduos que pertençam a apenas uma delas. Ao se fazer `equivalentTo`, se está indicando que todo elemento pertencente a A também pertence a B, **e vice-versa**. Sendo assim, para $\{1, 2, 3\} \in A$ e $\{4, 5, 6\} \in B$, `equivalentTo(:A :B)` fará com que tanto A quanto B sejam compostas pelos indivíduos $\{1, 2, 3, 4, 5, 6\}$.

Exemplificar com elementos do domínio escolhido na parte prática.

2. Compare a lógica descritiva que fundamenta a OWL 2 (na sua variação mais expressiva) com lógica de 1ª ordem. Apresente um exemplo do que é possível expressar com lógica de 1ª ordem que não conseguimos com lógica descritiva.

Solução:

4 Parte 2: Prática

1. Desenvolver um modelo conceitual (ontologia de domínio) utilizando OWL.

   Construam uma representação de domínio (utilizando *classes*, *relações* e *indivíduos*) de alguma área de interesse (pesquisa em computação, temas dos TCCs, filmes, esportes, músicas, hobbies, etc.).

- A avaliação considera principalmente a utilização dos axiomas de propriedades e de classes. Quanto mais restrições forem modeladas, melhor;
- Dentre as restrições, utilize no mínimo alguma de cardinalidade qualificada (`min`, `max`, ...) em alguma propriedade de objeto (ver na documentação da OWL, disponível no Moodle);
- Outro aspecto importante é a definição dos membros das classes. Utilizem suas definições para testar a definição dos axiomas e também para testar o funcionamento do mecanismo de inferência da linguagem.

Referências

[1] "Martin Kuba". "OWL 2 and SWRL tutorial". "`https://dior.ics.muni.cz/~makub/owl/`".