PEC3

Presentación

En esta Prueba de Evaluación Continuada se trabajan las aplicaciones de la representación del conocimiento a problemas reales.

La prueba se compone de tres actividades: el desarrollo de una aplicación que explota el conocimiento de una ontología, el uso de una ontología en OWL, y posibles aplicaciones donde se utilice el conocimiento representado en ontologías.

Competencias

En esta PEC se desarrollan las siguientes competencias del Grado en Ingeniería Informática:

- Capacidad de analizar un problema en el nivel de abstracción adecuado a cada situación y aplicar las habilidades y conocimientos adquiridos para abordarlo y resolverlo.
- Capacidad para aplicar las técnicas específicas de tratamiento, almacenamiento y administración de datos.

Objetivos

Los objetivos concretos de esta Prueba de Evaluación Continuada son:

- Saber utilizar el conocimiento almacenado en una ontología para resolver un problema concreto.
- Saber reconocer las técnicas de representación del conocimiento utilizadas en un sistema real.
- Saber proponer una solución a un problema concreto

Descripción de la PEC a realizar

Ejercicio 1 [40%]

Queremos crear un repositorio web con artículos de contenido médico. Una de las principales funcionalidades de esta web sería un buscador que haga uso de la semántica para identificar y devolver artículos relevantes. Este buscador tendrá que identificar los artículos relacionados con un conjunto de palabras clave entradas por el usuario. Los artículos relacionados no sólo serán aquellos que contengan las palabras clave, sino también los que que estén relacionados con las mismas.

Se pide que describáis los siguientes elementos:

1. Cuál es vuestra definición de artículo relacionado. Es una decisión importante que condicionará el resto del sistema. Por ejemplo, una definición podría ser que un artículo es relacionad si contiene palabras sinónimas de las palabras usadas en la búsqueda.
2. Qué fuentes de conocimiento (ontologías, tesauros, etc.) utilizarías. Busca si hay actualmente fuentes que exploren el ámbito de la medicina y os permitan llegar a vuestro objetivo.
3. Qué arquitectura tendría el sistema propuesto (componentes, software, software para almacenar y procesar las fuentes de información, etc.). Haz un pequeño esquema y explícalo.
4. ¿Cómo utilizarías las diferentes fuentes de información para devolver los documentos relacionados con el conjunto de palabras clave? Aquí sería bueno comentar dos aspectos:
    1. Definir un algoritmo básico que use las fuentes de información para refinar y expandir la búsqueda propuesta. No hay que describir el algoritmo con mucho detalle. El objetivo es que conozcáis las fuentes de conocimiento utilizadas y que la propuesta permita hacer un uso coherente de su información. Por lo tanto, esperamos algoritmos simples que muestren qué información utilizar, cómo y el por qué.
    2. Indicar si los artículos tendrán que estar etiquetados de alguna manera según las fuentes de información seleccionadas. En caso afirmativo indica cómo. En caso negativo justifica el porqué no.

Ejercicio 2 [30%]

En este ejercicio trabajaremos con una de las APIs que tenemos para tratar ontologías, y en especial, con su razonador. Para realizar el ejercicio usaremos el API de Jena. Esta API incluye varios razonadores que permiten generar nueva información a partir de un modelo y aplicar reglas propias.

Para realizar la actividad necesitareis el siguiente material:

- IDE Eclipse.
- Se proporciona un proyecto Eclipse con la configuración necesaria. Dentro del proyecto encontraréis:
  - La clase *Exercice2.java*, donde tendréis que completar el código donde hay los comentarios etiquetados con *//TODO*
  - Algunas funciones implementadas *(printStatements)* que podéis usar para resolver el ejercicio
  - En la carpeta data hay un fichero con la ontología a utilizar *(BabiesRDFComplete.owl*) y un fichero para llenar con las reglas que os pediremos (*babiesRules.rules*)
  - En la carpeta lib las librerías **Jena** y otras necesarias por el proyecto
- El manual que ofrece **Jena** en la página https://jena.apache.org/documentation/inference/ especialmente el apartado **The OWL Reasoner** en adelante.

En el documento se ha de entregar los fragmentos de código que corresponda y l**a salida que proporcione vuestro programa**.
Además **se tendrá que añadir el fichero Exercice2.java con vuestra implementación.**

Se pide implementar las siguientes partes:

a) Implementad la función *callReasonerOWL*:
1) Donde está el *TODO 1,* cread un modelo a partir del razonador y la ontología proporcionada.
2) Donde está el *TODO 2,* recuperad la instancia *Shmi* e imprimid por pantalla todos sus *statements*. Explicad si hay *statements* nuevos y por qué creéis que el razonador los ha añadido.
3) Donde está el *TODO 3,* y implementad el código para averiguar si *Shmi* es de tipo *GrandParent*. Sacar por pantalla el resultado de la validación.
4) Donde está el *TODO 4,* y implementad un código que diga si *Shmi* es *grandParentOf Luke*. Sacad por pantalla el resultado de la validación.

b) En este apartado se trabajará con reglas de derivación. Primero se añadirá alguna regla al fichero de reglas y después se completará el código fuente de la función *callOwnReasoner*:

1) Añadid al fichero *data/babiesRules.rules* dos reglas propias:
   a) Una regla que indique cuando una persona es abuela: *si a es hijo de b y b es hijo de c, c es abuelo de a*.
   b) Una regla que dispare una alerta (muestre un mensaje por pantalla) sobre los bebés de igual o menos de 48 meses que tienen sillita de coche asignada.
2) TODO 5: Implementad la llamada al razonador con las reglas propias que hemos generado.
3) TODO 6: Volved a implementar un código que diga si *Shmi* es *grandParentOf Luke*.
4) TODO 7: Implementad un código que diga si *Luke* es *grandChildOf Shmi*. Si se confirma, decid por qué se ha deducido esto.

Ejercicio 3 [30%]

En este ejercicio volvemos a utilizar el entorno anterior. Dentro del proyecto Eclipse os proporcionamos la clase *Exercice3.* Al igual que en el ejercicio anterior, esta clase contiene código etiquetado con TODO que se tendrá que implementar. Las medidas que se implementarán y las preguntas del ejercicio se basarán en el contenido del artículo *A Review of Semantic Similarity Measures in*WordNet . La lectura del último párrafo de la página 6, la página 7 y el principio de la página 8 (hasta el apartado 3.3) no será necesario para realizar el ejercicio.

Para realizar este ejercicio se recomienda dar un vistazo a la API de Jena (especialmente la clase *OntClass*):
https://jena.apache.org/documentation/javadoc/jena/index.html

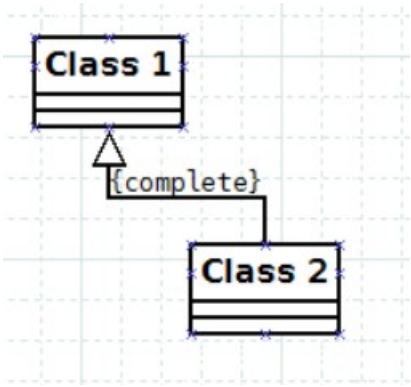
Además de mostrar los fragmentos de código en el documento de entrega **se tendrá que entregar el fichero Exercice3.java con vuestra implementación.**

a) Dado el fragmento de la ontología que se presenta en el artículo (figura 1), definir el valor de las medidas *deep_max* y nodo_max i el valor de las medidas *len*, *lso*, *depth*, hypo por los conceptos *bicycle* y girl .

b) Define el concepto de *information content* según tus palabras.

c) Teniendo en cuenta la estimación de *information content* realizada en la página 6 del artículo (fórmulas 8 a 9) y el esquema de la figura siguiente , ¿Cual de las tres igualdades/desigualdades siguientes es correcta? Justifica la respuesta (no hay que justificarla numéricamente, haciéndolo conceptualmente es suficiente).

- IC(Class1) < IC(Class2)
- IC(Class1) > IC(Class2)
- IC(Class1) = IC(Class2)



d) Describe con tus propias palabras las principales diferencias entre las medidas de similitud semántica basadas en caminos, en contenido de

información y en características. ¿En qué casos utilizarías cada una de ellas?

e) Implementad el código que falta para poder aplicar las medidas de similitud de caminos presentadas en las formulas 1 y 2 (la medida de camino mínimo y la medida de Wu & Palmer). De cara a los cálculos hay que tener en cuenta qué:
   - En la función *calculateDepth,* que calculará la profundidad entre dos nodos padre e hijo, debéis tener en cuenta que el razonador de Jena puede generar clases anónimas que hemos de ignorar cuando exploramos los padres o los hijos.
   - Podéis considerar que no hay herencia múltiple entre las clases que buscamos.
   - Tenéis la constante DEPTH_ONTOLOGY que dice cuál es la profundidad máxima de la ontología.
   - Para calcular la distancia entre dos nodos podéis utilizar como ayuda la función *calculateDepth* y la función que implementa Jena *OntTools.getLCA(m, c1, c2)* que calcula el mínimo padre común entre dos clases (c1 y c2).

f) Una vez implementadas las dos medidas, ejecutadlas en la ontología de prueba y comparad el resultados de los dos algoritmos, ¿qué os dicen?

Recursos

Los siguientes recursos son de utilidad por la realización de la PEC:

Básicos

- Módulo didáctico 1 y 2

Complementarios

- Tutoriales y APIO de Jena
  https://jena.apache.org/documentation/inference/
  https://jena.apache.org/documentation/ontology/#running-example-the-eswc-ontology
  https://jena.apache.org/documentation/javadoc/jena/index.html
- Artículo "A Review *of Semantic Similarity Measures in"* WordNet proporcionado con el enunciado
- Proyecto Eclipse con todas las librerías y ficheros necesarios.

Criterios de valoración

La ponderación de los ejercicios es la siguiente:

- Ejercicio 1: 40%
- Ejercicio 2: 30%
- Ejercicio 3: 30%

Formato y fecha de entrega

Hay que entregar **un único fichero .zip** que contenga:

1. el **documento** Word, Open Office o PDF con las respuestas a las preguntas
2. El **fichero Exercice2.java** con la implementación pedida al ejercicio 2, al lugares marcados por //TODO.
3. El **fichero Exercice3.java** con la implementación pedida al ejercicio 3, al lugares marcados por //TODO.

Este documento se tiene que entregar en el espacio de Entrega **y Registro de EC** del aula antes de las **23:59 del día 23 de mayo**. **No se aceptarán entregas fuera de plazo**.