TALLER No. 4B	Inferencias con OWL

Profesor: Jaime Alberto Guzmán Luna

Contenido del taller:

1. Detalles en el manejo de protege
2. Inferencias con OWL

Dominio de la ontología: Apartes de un árbol genealógico

PARTE 1: Individuos y consultas de la ontología

Abra la siguiente ontología y verifique los siguientes elementos en negro y complete los que están en rojo de la **TBox**:

Propiedades de la ontología de tipo "Object Property"

hasParent (inverse of isParent)
    hasFather (inverse of isFatherOf)
    hasMother (inverse of isMotherOf)
isParent
isFatherOf
isMotherOf

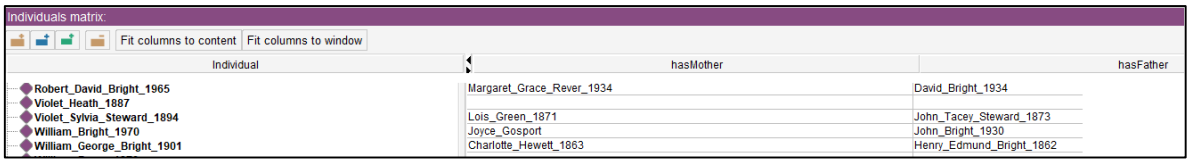
Tarea 1:
Encienda el razonador y responda: como queda la relación entre las propiedades isParent, isFatherOf, isMotherOf). Capturar la imagen inferida

Complete la ABox:
Verifique la existencia de los individuos desde la carpeta *Entities* y la subcarpeta *Individuals*.
¿Se dice que estos individuos son de tipo *named individual*, Por qué?
Complete el padre y la madre de los siguientes individuos:

Robert_David_Bright_1965
Violet_Sylvia_Steward_1894
William_Bright_1970
William_George_Bright_1901

Para ello utilice el complemento **Matrix** para Protégé en este momento. El plugin le permite agregar personas rápidamente en forma de una tabla regular y puede reducir significativamente el esfuerzo de agregar cualquier tipo de entidad a la ontología. Para instalar el plugin Matrix, vaya a *File » Check for plugins*. Seleccione el plugin " Matrix Views ". Haga clic en instalar, espere hasta que se confirme la instalación, cierre y vuelva a abrir Protégé. Vaya al elemento del menú " Window ", seleccione " Tabs " y agregue la " Individuals matrix ".

La tabla asociada a los anteriores individuos quedaría:



Para completar cada nombre basta con pararse en la casilla respectiva y digite la primera letra y luego *ctrl+barra_espaciadora* y aparecerán las opciones a elegir. Seleccione el respectivo individuo.

Tarea 2: Consulta en Lógica Descriptiva (DL Query):

La pestaña DL Query proporciona una función potente y fácil de usar para consultar en una ontología.

1. Ejecute el razonador.
2. Realice la consulta de DL hasFather value David_Bright_1934 y mire las respuestas (recuerde marcar la casilla de verificación respectiva en Protégé para incluir las *instances* en los resultados de su consulta).
3. Emita la consulta de DL isFatherOf value Robert_David_Bright_1965. Mire las respuestas.
4. Mire los hechos relacionados con Robert_David_Bright_1965.

PARTE 2: Ancestros y Descendientes

Implemente lo siguiente:

Propiedades de la ontología de tipo "Object Property"

hasRelation (symmetric)
hasAncestor (transitive, inverse of isAncestorOf)
isAncestorOf
Modificar la propiedad hasParent y ponerla como SubPropertyOf: hasAncestor

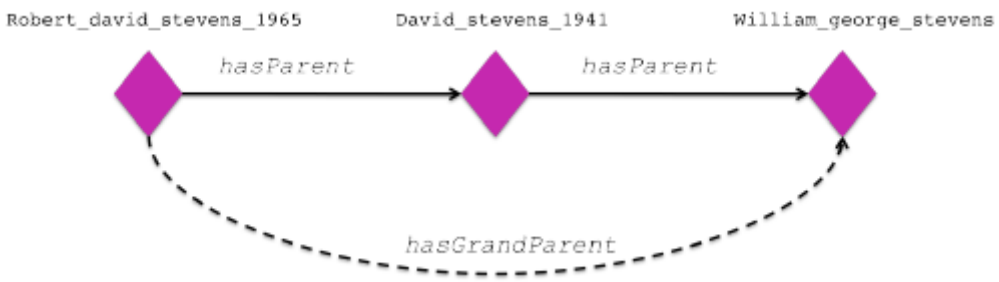
Tarea 3: DL Query:

1. Ejecute el razonador.
2. Realice la consulta hasAncestor value William_George_Bright_1901 y analice la respuesta (¿cuantas instancias genera la consulta?).
3. Realice la consulta isAncestorOf value Robert_David_Bright_1965 y analice la respuesta.
4. Responda la siguiente pregunta: ¿Por qué es importante en los resultados anteriores que hasAncestor sea transitive?

Propiedades de la ontología de tipo "Object Property"

Haremos uso de la propiedad: **owl:propertyChainAxiom** que maneja el tema de propiedades encadenadas

Se define la propiedad hasGrandparent asi:



Entonces
hasGrandparent (SubPropertyOf: hasAncestor, **propertyChainAxiom: hasParent o hasParent**, inverseOf isGrandparentOf)
hasGrandmother (subPropertyOf :hasGrandparent, inverseOf: isGrandmotherOf,
propertyChainAxiom: hasParent o hasMother)
hasGrandfather (subPropertyOf :hasGrandparent, inverseOf: isGrandfatherOf,
propertyChainAxiom: hasParent o hasFather)
isGrandparentOf
isGrandmotherOf
isGrandfatherOf

Tarea 4: DL Query:

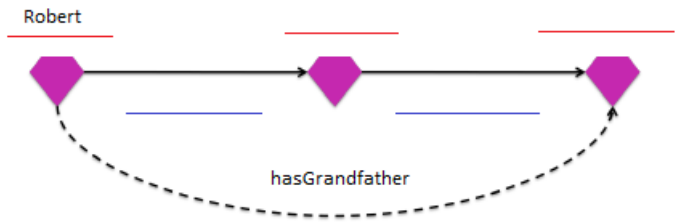
1. Vaya a la pestaña de individuals e inspeccione las Property Assertions de Robert_David_Bright_1965

Cuantas aserciones tiene Robert?

2. Ejecute el razonador. Complete los siguientes espacios asociado al razonamiento frente a las relaciones que aparecen para Robert que permiten cumplir la Property **hasGrandfather**. Dado que en la figura inicial de este apartado se muestra el caso de que Robert tiene como abuelo a William, realice a continuación el razonamiento para el otro abuelo que Robert tiene. (Escribir en la línea solo el nombre principal del familiar).

a. Robert cumple la propiedad **hasMother** con _____.
b. Por implicación dado que **hasParent** es super-propiedad de **hasMother,** Robert cumple con la propiedad **hasParent** con_____.
c. Igualmente, _____ cumple con la propiedad **hasFather** con _____.
d. Entonces, dado la subproperty chain hasGrandfather implica que _____ cumple con la propiedad hasGrandfather con _____.

3. Complete el siguiente gráfico de manera similar como el original, pero para **hasGrandfather.** En líneas rojas solo los nombres y en las líneas azules las relaciones.



4. Realice la consulta hasGrandfather value William_George_Bright_1901 y analice la respuesta. ¿Cuantas instancias se generan?

PARTE 3: Modelado de la clase Persona

A. Implemente lo siguiente para definir las clases Person y Sex:

Clases de la ontología

Subclasses of Thing

Person
Sex

Subclasses of sex

Maleness
Femaleness

Hacer disyuntas las siguientes clases:

Sex y Person
Maleness y Femaleness

Propiedades de la ontología de tipo "Object Property"

hasSex (domain: Person, range: Sex, Functional)

Adicionar a la clase Person la siguiente restricción: (hasSex some Sex)

Implementar las siguientes restricciones de equivalencia

Sex	Maleness or Femaleness

B. Implemente lo siguiente para definir Man y Woman :

Clases de la ontología

Subclasses of Thing

Man
Woman

Implementar las siguientes restricciones de equivalencia

Man	Person and (hasSex some Maleness)
Woman	Person and (hasSex some Femaleness)

Tarea 5: Inferencia:

1. Observe como las clases Man y Woman se situan en la jerarquía de clases:
   Man y Woman están o no están al mismo nivel que Person?
   Analice y escriba el porqué de su respuesta.

Nota: en la versión actual de protegé la jerarquía que se genera con las 2 restricciones de equivalencias anteriores se muestra igual sin el uso del razonador como con el razonador encendido. Lo normal es que solo se mostrara cuando el razonador se encendiera y se mostrara lo inferido.

C. Implemente lo siguiente para definir el parentesco:

Se describe el parentesco de un objeto de persona. En este dominio cada una de las personas tienen una madre y un padre. Aquí en este dominio estamos hablando solo de padres y madres biológicos (no se trata el tema de la adopción).

Clases de la ontología

Incluir las restricciones en rojo a la clase Person la cual se veria asi:

Class: Person
SubClassOf: (hasFather some Man), (hasMother some Woman), (hasSex some Sex)
DisjointWith: Sex

Propiedades de la ontología de tipo "Object Property"
Modificar las siguientes propiedades existentes así:

hasMother (domain: Person, range: Woman)
hasFather (domain: Person, range: Man)
hasParent (domain: Person, range: Person)

Tarea 6: DL Query:

1. Con el razonador encendido, realizar consultas de DL para Person, Man y Woman; mire las respuestas y cuente los números de individuos bajo cada clase;

Person:_____
Man:_____
Woman:_____

¿Cuantos individuos no tienen sexo (es decir no son Man ó Woman) y escriba el por qué (sugerencia: analice las relaciones hasFather y hasMother)?

_____

_____

¿Si observas el número de individuos named (original), por qué no aparecen todos como individuos de Person??_____

D. Asignar Domains y Ranges a las Properties del dominio:

1. Asegurarse que las clases Person, Man y Woman son domains y ranges para hasFather, hasMother y hasParent. Nota los **azules** no se deben implementar ya que se heredan y los **verdes** se implementaron en el numeral anterior y son redundantes, pero no afectan la definición del dominio y se pueden dejar. El anaranjado se explica en la Nota2:

hasFather (Domains: **Person**, Ranges: Man)
hasMother (Domains: **Person**, Ranges: Woman)
hasParent (Domains: **Person**, Ranges: **Person**)
hasGrandfather (Domains: **Person**, Ranges: Man)
hasGrandmother (Domains: **Person**, Ranges: Woman)
hasGrandParent (Domains: **Person**, Ranges: **Person**)
hasAncestor (Domains: **Person**, Ranges: **Person**)
hasRelation (Domains: **Person**, Ranges: Person)

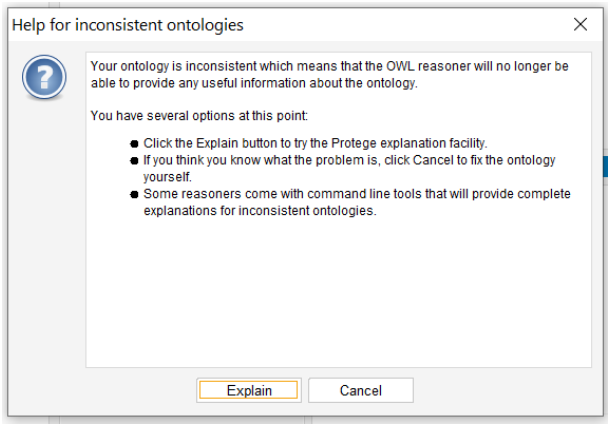
Nota1: Protégé, en su versión actual no visualiza dominios y rangos heredados de la misma manera que muestra relaciones inversas inferidas. Por lo tanto, en realidad la herencia en dominios y rangos si existe, pero no se visualiza.
Nota2: En el caso de **hasRelation** no es necesario implementar Domains: **Person** porque esta propiedad es Symetric y automáticamente por razonamiento se le asigna Domains: **Person** desde su **Ranges:Person**.

Tarea 7: Inconsistencia:

Para comprobar la herencia de los Domains y Ranges realice lo siguiente:

a. Quite en **hasRelation** el que sea Symetric (con esto la inferencia no le asigna Domains: Person).
b. Cree una instancia **sex1** de la clase **Sex** (recuerde que la Clase Sex es disjoint con Person)
c. Cree la siguiente Property assertion para sex1: sex1 **hasAncestor** David_Bright_1934.
d. Ejecute el razonador y verifique que **NO** sucede ninguna inconsistencia.
e. Pare el razonador.

f. Nuevamente asigne a **hasRelation** el que sea Symetric.
g. Ejecute nuevamente el razonador y verifique que sucede una inconsistencia y aparece el siguiente Cuadro (darle cancel en dicho cuadro):



¿Porque se da esta inconsistencia? (ver la imagen que sigue para entender la respuesta de esta pregunta) _____
_____

h. Detenga el razonador.
i. Borre la instancia sex1 con su Property assercion.
j. Encienda nuevamente el razonador para verificar que ya no existe inconsistencias.

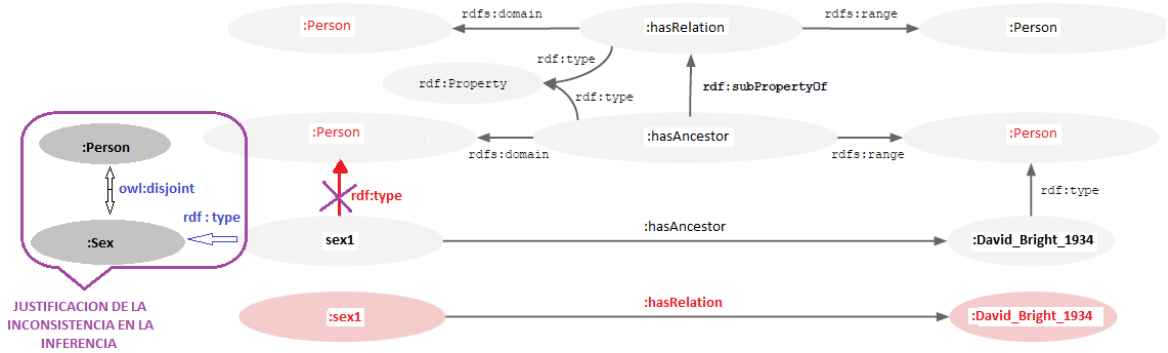
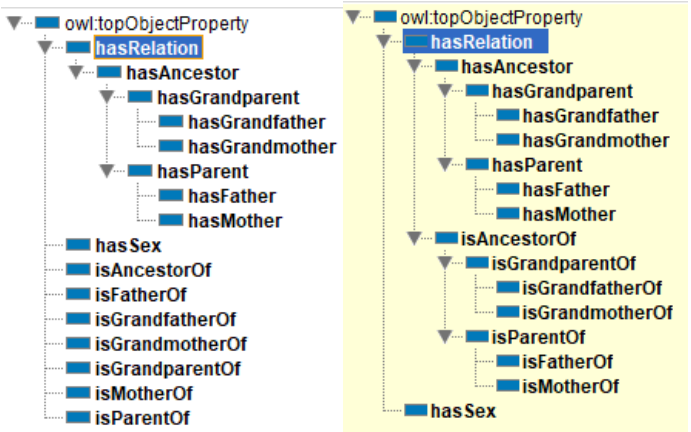


Imagen con el análisis de la inconsistencia

Con el fin de verificar el estado, así se vería el esquema de las Object Property



Tarea 8: Inconsistencia:

Para comprobar la herencia de los Domains y Ranges realice lo siguiente:

a. Adicione que Robert_David_Bright_1965 hasMother Iris_Ellen_Archer_1906
b. Ejecute el razonador.
c. Realizar la siguiente DL Query: hasMother value Iris_Ellen_Archer_1906
d. Mire las respuestas y responda lo siguiente:
   Porque aparece que Richard_John_Bright_1962 cumple con esta consulta?
   _____

e. Apague el razonador
f. Incluya que todos los individuos son sean diferentes entre ellos mismos (**owl:distinctMembers)**, en protege esto se hace con Different Individuals:

g. Encienda el razonador. Verifique que ocurre una inconsistencia. Oprima el botón explain. ¿Acorde a la información mostrada en dicha pantalla en sus propias palabras porque ocurre esta inconsistencia?
   _____
   _____
h. Apague el razonador
i. Borre la afirmación incluida en el numeral a.
j. Encienda el razonador y verifique si pasa lo mismo que en el numeral g.

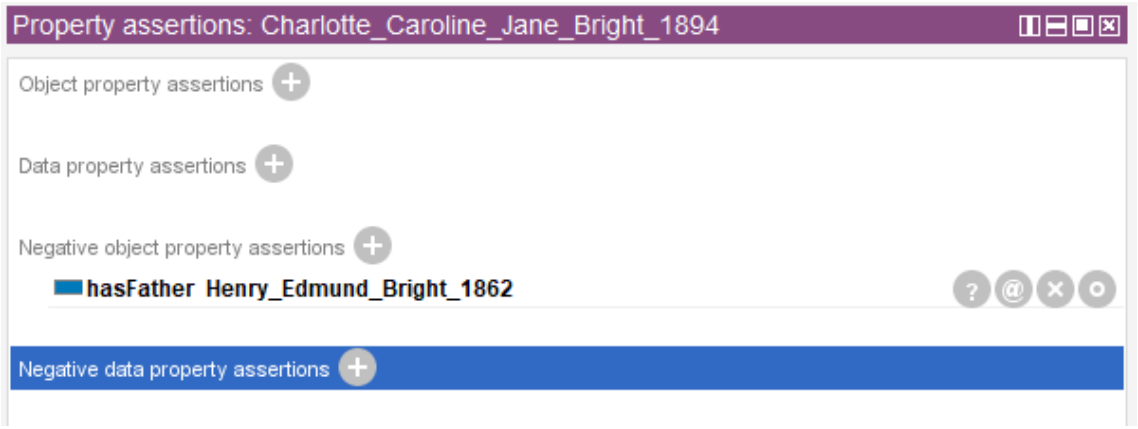
PARTE 4: Propiedad Negada

Implemente lo siguiente:

NegativeObjectPropertyAssertion (característica del OWL2)

Implementar la siguiente afirmación: “Charlotte_Caroline_Jane_Bright_1894 nunca quiere tener como papa a Henry_Edmund_Bright_1862”.

Ir a la pestaña de individuos y en la sección de Property assertions escribir en Negative Object Property Assertion lo siguiente: hasFather Henry_Edmund_Bright_1862


Property assertions: Charlotte_Caroline_Jane_Bright_1894
Object property assertions
Data property assertions
Negative object property assertions
hasFather Henry_Edmund_Bright_1862
Negative data property assertions

Referencias

- **Manchester Family History Advanced OWL Tutorial. Edition 1.1**
- **[**http://mowl-power.cs.man.ac.uk/fhkbtutorial/resources/FHKB-tutorial_v1_1.pdf**]**