Editing e applicazioni delle ontologie formali

Strumenti di editing

Strumenti per il ragionamento automatico

Ragionamento in RDF

La rappresentazione dei dati in RDF permette di effettuare alcuni tipi di ragionamento
Per esempio dalle triple:

ex:bob foaf:knows ex:alice .	*(Bob conosce Alice)*
foaf:knows rdfs:domain foaf:Person .	*(la proprietà conoscere ha come dominio la classe Person)*

Si può derivare che Bob appartiene alla classe Person (ex:bob rdf:type foaf:Person) dato che il dominio della proprietà knows è vincolato a Person

Oppure, è possibile verificare che la seguente tripla è inconsistente con le definizione dellla proprietà ex:age

ex:bob ex:age "forty"^^xsd:integer . *(Bob ha "quaranta" anni)*

Dato che quest'ultima è vincolata ad avere un intero come valori (e non una stringa)

Editor di ontologie

- L'editor di ontologie maggiormente diffuso è Protégé:
- http://protege.stanford.edu
- Attraverso l'editor gli utenti creano l'ontologia utilizzando menu e maschere predefinite, senza conoscere la sintassi esatta del linguaggio in cui l'ontologia verrà codificata dall'editor
- L'editor contiene alcuni plugin indispensabili:
  - Per visualizzare l'ontologia in un formato interamente grafico
  - Per effettuare il ragionamento automatico, necessario durante lo sviluppo per assicurarsi che l'ontologia sia logicamente consistente

Struttura dell'editor

- L'interfaccia di Protégé mostra un insieme di etichette (tab) che permettono di passare da una funzionalità all'altra.
- Il riferimento principale è la tassonomia delle classi dell'ontologia
  - Entities mostra tutte le entità (classi e individui) creati
  - Classes permette di operare sulle classi (T-box)
  - Individuals permette di operare sulle istanze singole (A-box)
- Se il ragionatore (reasoner) è attivato, per ogni tab si può vedere l'ontologia arricchita con i risultati del ragionamento automatico
  - In questo caso, gli elementi aggiunti via inferenza saranno evidenziati in giallo.

Immagine editor dopo ragionamento

Creare classi

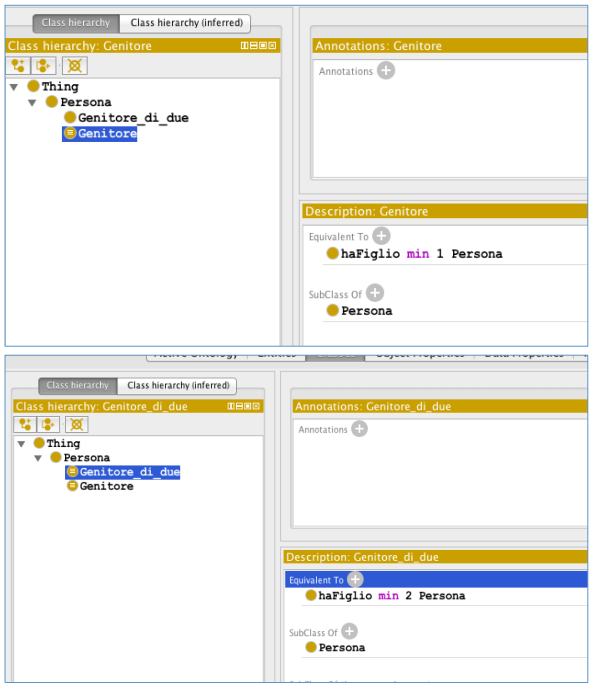
- L'ontologia vuota contiene sempre una classe top-level, chiamata Thing
- Una volta selezionato il pannello che rappresenta le classi, si crea una classe come "sorella" o come "figlia" di una classe esistente
- I pannelli sulla destra dell'interfaccia permettono di descrivere le classi
  - Contengono informazioni su classi sovra-ordinate o sotto-ordinate, vincoli sulla classe, disgiunzioni tra classi, ecc. che possono essere modificate

Creare proprietà

- Nei tab properties un pannello posto a sx mostra la gerarchia delle proprietà definite
- Anche qui è presente una proprietà standard, top level (TopLevelProperty)
  - In molti casi non è necessario creare sotto-proprietà, tutte le proprietà possono stare allo stesso livello
- I pannelli a dx mostrano le caratteristiche della proprietà selezionata:
  - Proprietà più specifiche/generali, dominio e range, caratteristiche della proprietà, ecc.
  - Queste caratteristiche sono editabili
  - Se non specificato dominio e range, la proprietà si applica a tutte le classi

Esempio: ragionamento su classi

- Creiamo due classi "sorelle" di cui una sussume logicamente l'altra
- Il reasoner colloca correttamente la classe sussunta come sottoclasse di quella più generale

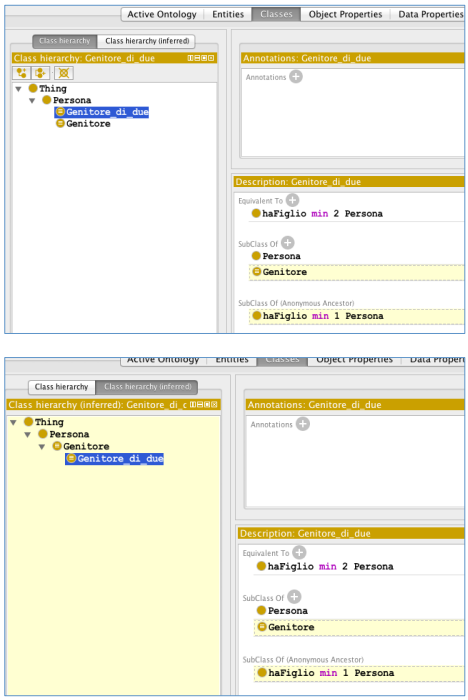


La proprietà haFiglio ha come dominio Persona e come range Persona

La classe Genitore è vincolata a avere almeno un figlio

La classe Genitore_di_due è vincolata a averne almeno due

Le due classi sono definite come sottoclassi di Persona.



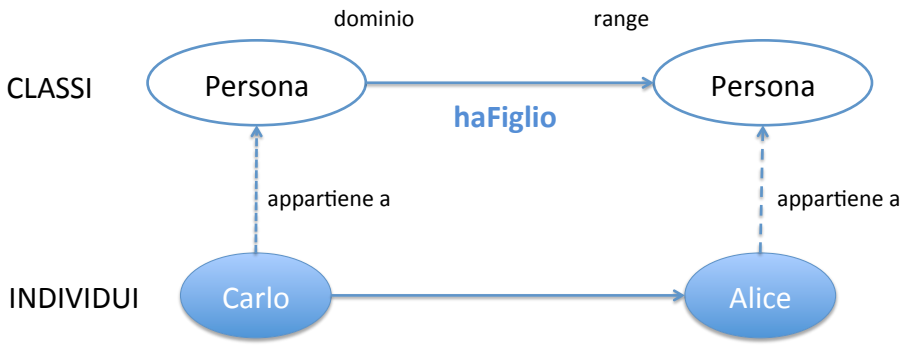
Il reasoner inferisce che la classe Genitore_di_due è sottoclasse di Genitore

Nel pannello delle classi, la gerarchia inferita mostra che la classe Genitore_di_due è stata spostata sotto la classe Genitore

Inserimento di individui

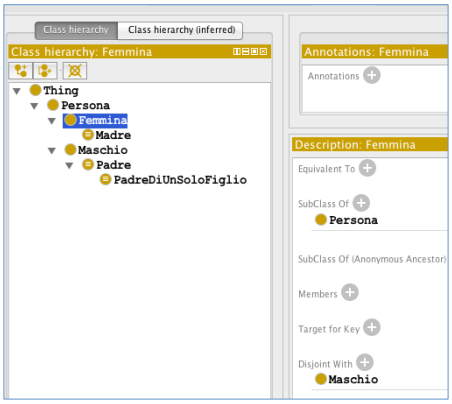
- Gli individui vengono inseriti direttamente in una classe
- Dopo averli inseriti è possibile predicarne le caratteristiche, cioè specificarne le proprietà che li mettono in relazione con altri individui o con un dato
- Il ragionamento può eventualmente ricollocare un certo individuo in un'altra classe

Relazione classi - individui



Esempio: disgiunzione

- Colloco lo stesso individuo in classi disgiunte
- Il reasoner mi segnala che l'ontologia è incosistente

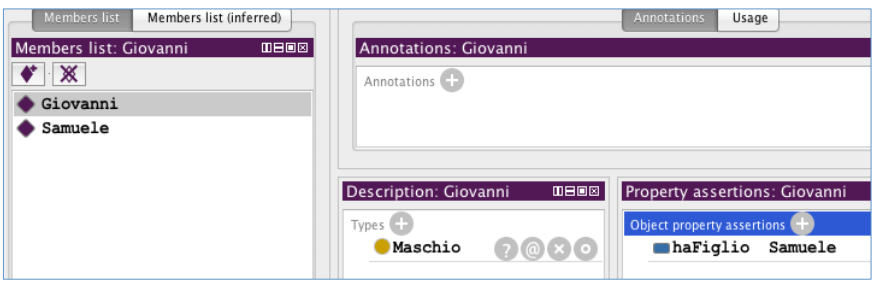


Dato che le classi Femmina e Maschio sono disgiunte, non posso collocare lo stesso individuo in entrambe

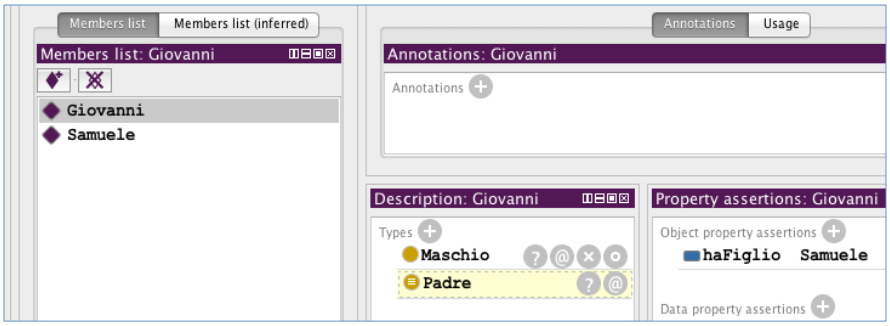
Reason for inconsistency: individual Giovanni is forced to belong to class "Maschio" and its complement

Inserimento di individui

- Inserendo un individuo con le caratteristiche di una certa classe, l'individuo viene classificato come appartenente alla classe dal reasoner
- Anche se è stato collocato manualmente in una classe più generale



Classificazione automatica

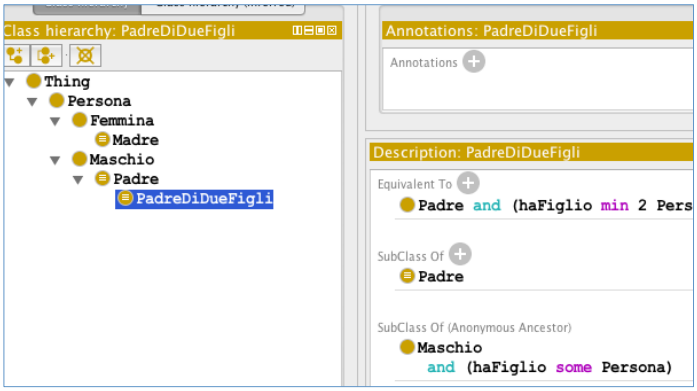


- L'individuo Giovanni è stato spostato nella classe Padre (classe più specifica di Maschio in cui era stato collocato
  - Perché Giovanni "haFiglio" Samuele
- L'inferenza scatta perché la classe Padre è una classe *definita*, a cui sono associate condizioni necessarie e sufficienti

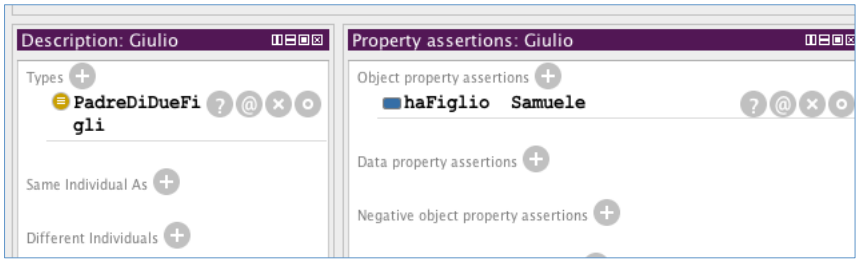
Assunzione di mondo aperto

- Le inferenze effettuate dal reasoner possono sembrare poco intuitive in alcuni casi
- La maggior parte dei sistemi di ragionamento automatico segue l'assunzione di mondo chiuso:
  - Ciò che non è rappresentato esplicitamente nel sistema viene assunto falso
  - Provate a salire su un volo se non siete nella lista dei passeggeri
- Il ragionamento sulle ontologie OWL segue invece l'assunzione di mondo aperto:
  - Il fatto che un'informazione non sia rappresentata nel sistema non determina che essa sia assunta falsa.

Esempio



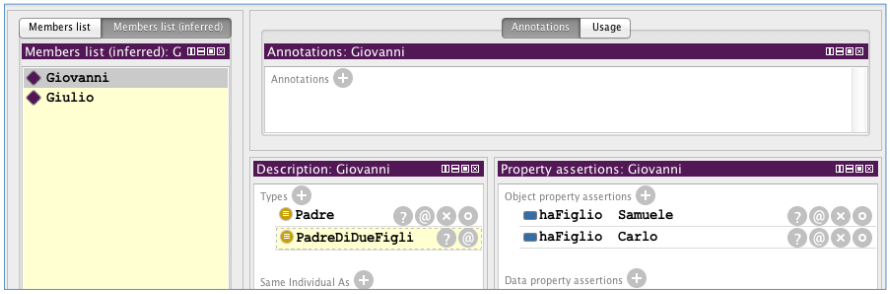
La classe PadreDiDueFigli è una classe definita su cui vale il vincolo che per i suoi appartenenti la relazione haFiglio abbia almeno due individui diversi



Il fatto che un certo individuo, Giulio, sia padre di un solo figlio, non genera una inconsistenza, anche se Giulio è stato collocato nella classe PadreDiDueFigli

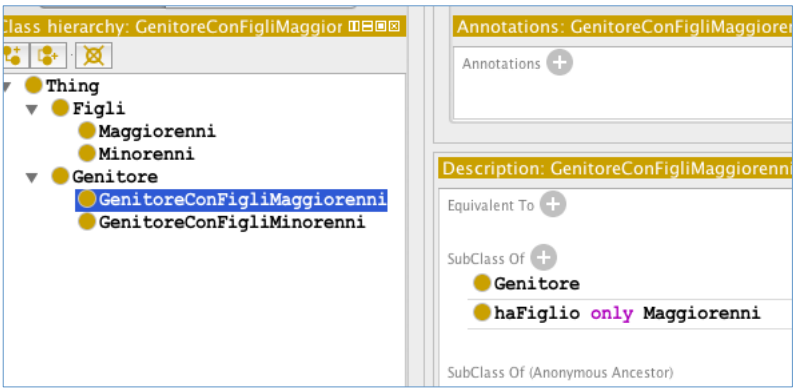
Il ragionatore infatti, applica l'assunzione di mondo aperto: non viene asserito che Giulio non abbia altri figli e quindi l'appartenenza alla classe, se asserita esplicitamente, non è in conflitto con le proprietà dell'individuo Giulio

Esempio 2



Il reasoner inferisce correttamente che Giovanni, in quanto padre di due Samuele e Carlo, è appartiene alla classe PadreDiDueFigli
Ma solo se Samuele e Carlo sono stati specificati esplicitamente come due diversi individui.
Altrimenti (mondo aperto) potrebbero essere lo stesso individuo e quindi il vincolo non sarebbe soddisfatto.

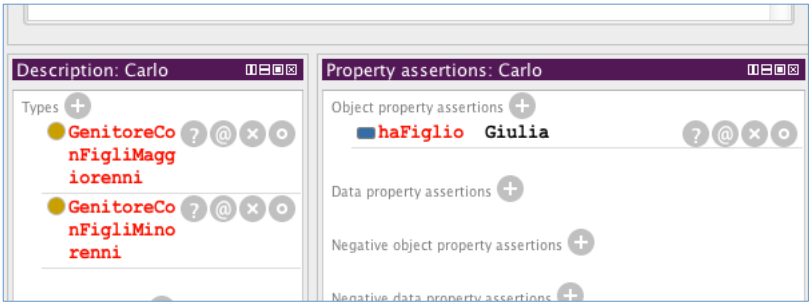
Quantificatore universale



Le classi GenitoreConFigliMaggiorenni e GenitoreConFigliMinorenni non sono disgiunte, ma lo sono le classi Maggiorenni e Minorenni

Se un individuo viene collocato in entrambe le classi, il reasoner rileva una inconsistenza

Esempio



Se Carlo, che haFiglio Giulia (minorenne), viene dichiarato come membro di entrambe le classi, il reasoner rileva un'inconsistenza.

Giulia infatti si troverebbe a appartenere a due classi distinte, che sono state dichiarate come disgiunte.

Modellazione

- Dominio ristoranti
- Esistono ristoranti e tipi di cucina
- I tipi di cucina sono: Giapponese, Cinese, Thai, Piemontese, Toscana
  - Per ogni tipo ci sono singole "cucine": noodles e curry per Thai; Fiorentina e Livornese per Toscana, ecc.
  - Thai, Giapponese e Cinese sono cucine orientali
  - I tipi di cucina sono tutti disgiunti tranne la cucina di pesce
- Ci sono una serie di ristoranti
- I ristoranti hanno una cucina di un certo tipo
- La classe "ristoranti orientali" è quella dei ristorantni che hanno cucina orientale

Inserimento di individui: esempio 2

- Violazione di vincoli sulle proprietà
- Data properties

- Object properties (una esclude l'altra)