Domande Esame

Domanda #1

Domanda: Si dia una descrizione della frame theory e si definisca la struttura di un frame.

La frame theory è un formalismo per la rappresentazione della conoscenza strutturata, che rappresenta un'evoluzione rispetto al formalismo delle reti semantiche, in cui viene aggiunta la possibilità di rappresentare conoscenza di tipo stereotipico. Come suggerisce il nome per l'appunto, il formalismo si basa sul concetto di *Frame*, una struttura dati atta a rappresentare delle conoscenze stereotipiche riguardo a determinate situazioni. Un frame è composto da una serie di slots organizzati in livelli, che vengono riempiti di valori in base alla situazione specifica. Gli slots a livello più alto rappresentano della conoscenza steoreotipica della situazione, per cui contengono valori di default che sono assunti essere sempre veri rispetto alla situazione. Quelli più bassi, invece, possono essere riempiti con dati o istanze specifiche della situazione. I valori degli slots possono essere sia dati, che riferimenti ad altri frames. Inoltre, agli slots possono essere associate anche delle procedure di calcolo dei valori (*if-needed*), per cui si ottiene sia una formalizzazione sia a livello di conoscenza, che a livello di procedure di calcolo per ottenerla. Più nel dettaglio, la struttura di un frame è composta dai seguenti elementi:

- **Identificativo**: permette l'identificazione del frame da parte di altri frames;
- **Slot generali/specifici**: permettono di creare una tassonomia di frame;
- **Slot generici**: rappresentano la conoscenza stereotipica;
- **Procedure per il calcolo di valori**: permettono di calcolare automaticamente alcuni valori degli slots sulla base di condizioni specifiche;
- **Valori predefiniti**: valori impostati di defaults.

Domanda #2

Domanda: Si elenchino le differenze tra un grafo relazionale (rete semantica) e una rete proposizionale.

Siccome le reti proposizionali sono un'evoluzione delle reti semantiche, ci sono molte similitudini tra i due formalismi di rappresentazione. Nonostante ciò, esistono alcune differenze. La prima è che una rete proposizionale è in grado di rappresentare conoscenza di tipo *epistemico* (credenze soggettive) per mezzo dei nodi proposizionali. Mentre le reti semantiche non hanno questa possibilità, per cui la conoscenza asserita dal grafo è in qualche modo di tipo *eterarchico*, cioè tutta allo stesso livello di importanza. La seconda è che introducendo quantificatori, connettivi e variabili le reti semantiche proposizionali permettono di ottenere lo stesso livello di espressività della logica del prim'ordine, mentre le reti semantiche ne implementano solo un sottoinsieme.

Domanda #3

Domanda: Quali sono le primitive di SnePS?

- `assert`: asserisce della conoscenza nella rete che viene creduta vera, ad esempio il comando `assert(member Tweety class Canary)` asserisce che "Tweety" è un canarino.
- `build`: permette di rappresentare della conoscenza che non viene creduta dal sistema ma che comunque è presente.
- `find`: cerca uno o più nodi della rete. Permette di utilizzare la rete come un database di fatti che può essere interrogato descrivendo il tipo di fatti cercato. (es. `find(class elephant)`)
- `deduce`: permette di cercare proposizioni che non sono direttamente asserite dal sistema, ma che esso è comunque in grado di inferire. (es. `deduce(member $x class canary)` cerca tutti gli `$x` che appartendono alla classe dei canarini).
- `define-path`: crea una relazione a partire da un percorso di relazioni (crea di fatto una *property-chain*), componendole col comando `compose`.

Domanda #4

Domanda: Quali sono i tipi di inferenze possibili con SnePS?

Ci sono 3 tipologie di metodi per fare inferenze su SnePS:

1. **Riduzione**: consiste nel dedurre da un grafo una porzione contenuta in esso. (es. se diciamo che John da un libro a Mary, allora possiamo anche dedurre (credere) la proposizione che John stia dando un libro.)
2. **Basate su percorsi**: consiste nello stabilire che un percorso fatto da certe relazioni sia uguale ad una singola relazione.
3. **Basato su regole**: sono regole del tipo antecedente-conseguente che possono essere definite dall'utente in modo arbitrario, e fanno parte della base di conoscenza stessa. Permettono di aumentare la conoscenza della rete non implicita e sono *domain-specific*.

Domanda #5

Domanda: Descrivere cos'è e in cosa consiste il formalismo SWRL.

Il *Semantic Web Rule Language* è un linguaggio per la definizione di regole nel web semantico. Tali regole permettono sia di inferire nuova conoscenza (regole dichiarative) che di effettuare delle azioni (regole di produzione). È bene notare che le regole di produzione non possono creare nuovi individui, ma solo asserire nuove *relazioni/proprietà* tra le entità già presenti nel grafo. L'impiego di queste regole permette di definire ragionamenti specifici del dominio di riferimento, senza dover intaccare l'architettra dell'ontologia. Le regole SWRL sono composte da un antecedente e un conseguente in cui l'antecedente viene valutato sul grafo

che rappresenta l'A-Box e la T-Box. Sia l'antecedente che il conseguente sono composti da congiunzioni di atomi che possono essere:

- `C(x)`: `x` appartiene alla classe `C`
- `P(x, y)`: `x` è legato a `y` tramite la proprietà `P`
- Operatori built-in quali:
  - `sameAs(x, y)`: `x` è uguale a `y`
  - `differentFrom(x, y)`: `x` è diverso da `y`

Esempio di regola:

```
haRuolo(?a, ?r) ^ haCompenso(?r, ?c) ^ swrlb:greaterThan(?c,
30000) -> AttoreMoltoPagato(?a)
```

Domanda #6

Domanda: Descrivere in maniera informale gli assiomi di SUMO (navigare l'interfaccia).

È possibile notare dall'interfaccia mostrante i risultati della ricerca, che ogni synset di WordNet è associato ad un identificativo ed una breve descrizione. Ad ogni synset di WordNet, corrisponde un diverso concetto di SuMO. Un concetto di SuMO può corrispondere esattamente al significato del synset (*equivalent mapping*), oppure il concetto SuMO include al suo interno come concetto più specifico, anche il significato del synset (*subsumption mapping*). Si noti come una parte del significato presente nel linguaggio naturale rispecchiato nella distinzione dei synset di WordNet, non viene colta dall'ontologia, che ha per necessità una granularità maggiore (es. vari significati WordNet di shout vengono mappati in un unico concetto SuMO). Ne consegue che vari synset diversi possono corrispondere allo stesso concetto SuMO.

Domanda #7

Domanda: Descrivere l'allineamento tra WordNet e Wikipedia (YAGO).

YAGO è una risorsa ontologica costruita su Wikipedia, WordNet e GeoNames. Il mapping è stato effettuato a partire da Wikipedia, per mezzo della seguente metodologia. Partendo dalla pagina Wikipedia, si selezionano le categorie, considerandone solamente la testa del nome (parola reggente). Se è plurare propone la categoria come classe e l'entità come membro della classe, collegandola al synset WordNet più frequente (euristica empiricamente accettabile). Successivamente, un insieme di pattern definiti a mano mappano le categorie e gli attributi della infobox su un insieme di template. Inoltre possono essere utilizzati alcuni set di regole per migliorare quanto estratto:

- Regole specifiche per migliorare la conoscenza linguistica.
- Implication rules: se nella KB compare un certo fatto, deve esserne aggiunto un altro.
  - eg Se $A\, property1\, B \implies A\, property2\, B$.

- Factual rules: collegamento tra le categorie di Wikipedia al synset WordNet.
  - eg $capital\ hasPreferredMeaning\ wordnet-capital-108518505$.
- Replacement rules: se un dato segue un certo pattern, rimpiazzarlo con un altro.
  - eg *USA* $\rightarrow$ *United States of America*.
- Extraction rules: se una parte del testo sorgente segue un certo pattern, si aggiungono determinati fatti.
  - eg *births* $\rightarrow$ *wasBornOnDate*.

Domanda #8

Domanda: Quali sono le possibili serializzazioni di RDF?

RDF è solo una specifica formale per cui presenta una molteplicità di serializzazioni:

- Turtle:
  - Prefissi definiti con `@prefix rdf: <...>`
  - Ogni tripla termina con il punto `.`
  - Se una tripla ha lo stesso oggetto allora vengono separati i predicati con `;`
  - Il blank node si rappresenta con `[]`
  - Se più triple hanno come stesso soggetto un blank node, allora si possono raggruppare (separate da `;`) con delle parentesi quadre
- RDF-XML;
- N-Triples: sottoinsieme di Turtle senza abbreviazioni e senza prefissi.
- N-Quadruples: è come N-Triples ma integra nella tripla anche il graph-IRI a cui fa riferimento
- Json LD;
- N3;
- RDFA.

Domanda #9

Domanda: Che cos'è un blank node?

I blank node sono nodi che fungono da variabile e permettono di descrivere qualcosa senza doverlo nominare esplicitamente in un grafo di conoscenza.

Ad esempio, consideriamo la madre di John. Potremmo sapere che è spagnola e abile in cucina, ma non abbiamo altri dettagli. In RDF, possiamo rappresentare queste informazioni utilizzando un nodo vuoto:

```
:john :haMadre [ :haNazionalità "spagnola" ; :bravaIn :cucina ]
```

Domanda #10

Domanda: Che differenza c'è tra TBox e ABox?

Le basi di conoscenza descritte tramite OWL (o con logiche descrittive) differenziano tra due tipologie di conoscenza:

- **T-Box**: parte della base di conoscenza che consiste nella *terminologia* che costituisce la conoscenza generale del dominio descritto dalla risorsa.
- **A-Box**: comprende tutte le asserzioni che descrivono entità concrete o specifiche del dominio dato.

Domanda #11

Domanda (importante): Quali sono le differenze tra *property-centric* (RDFS) e *class-centric* (OWL)?

RDFS è un linguaggio di modellazione dei dati property-centric. Si dice property-centric perché permette di asserire proprietà che legano le entità con altre risorse. Tramite esso è possibile definire se una risorsa è una classe, una proprietà, se è un'istanza di una determinata classe, se una proprietà è una sotto-proprietà di un'altra proprietà e, infine, il range e il domino di una proprietà. Non si può però esprimere che due classi sono disgiunte, che una proprietà è transitiva oppure che esistono individui. Inoltre, il ragionamento automatico è ridotto ad una serie di regole di inferenza molto primitivo. OWL nasce per superare queste limitazioni, per cui viene detto *class-centric*, poiché tramite esso è possibile definire proprietà, classi, individui e letterali. Siccome OWL si basa sulle logiche descrittive computazionali, è possibile fare ragionamento automatico sulle basi di conoscenza scritte in questo linguaggio (a differenza di RDFS che permette un tipo di ragionamento molto più primitivo).

Domanda #12

Domanda: Cos'è un named graph?

Un named graph è un grafo RDF identificato da un URI che contiene un insieme di triple RDF associate a quel grafo specifico. I named graph consentono di organizzare i dati in modo strutturato e di attribuire significato e contesto specifici a un insieme di triple. Possono essere utilizzati per rappresentare diverse fonti di dati, punti di vista o contesti.

Importante è anche il default graph, cioè il grafo RDF principale che contiene tutte le triple RDF non associate a un grafo specifico. È il grafo con cui il data store di riferimento esegue le query SPARQL in caso non venga specificato un grafo specifico. Può essere considerato come il grafo predefinito nel quale le informazioni sono contenute se non sono associate a un grafo nominato.

Domanda #13

Domanda: Ontology Engineering (NEON e OntoClean).

L'*Ontology Engineering* è l'ambito che si occupa del design delle ontologie e del loro mantenimento. L'idea è quella di fornire principi di modellazione che siano

standard e comuni a tutte le ontologie in modo da facilitarne l'interoperabilità. Le due principali metodologie viste sono OntoClean e NeOn.

OntoClean consiste in un'*analisi* delle classi dell'ontologia nei termini delle loro metaproprietà, che può portare alla verifica e all'eventuale *revisione* della struttura tassonomica dell'ontologia. Le metaproprietà menzionate sono 4, ognuna identiticata da una lettera:

- (I) Identità - Ciò che è identificabile
- (U) Unità - Ciò che è unitario
- (R) Rigidità - Ciò che non è soggetto a cambiamenti
- (D) Dipendenza

La metodologia consiste nei seguenti passaggi:

1. Ogni classe viene annotata con le meta-proprietà:
   - `+P` se la classe possiede la proprietà `P`;
   - `-P` se la classe non possiede la proprietà `P`;
   - `~P` se la classe possiede l'anti-proprietà di `P`.
2. Si analizzano le meta-proprietà delle classi, da cui emergeranno determinati vincoli di sussunzione (struttura tassonomica) in base alle sue metaproprietà (es. Una classe *anti-rigida* non può sussumere una classe *rigida*).
3. I vincoli si utilizzano per ristrutturare e verificare la gerarchia delle classi.

NeOn consiste invece in una metodologia orientata agli aspetti collaborativi dello sviluppo e mantenimento di un network di ontologie. Prevede un insieme di 9 scenari possibili a cui sono associate specifiche attività e documenti:

- Scenario 1: Dalle specifiche all'implementazione: prevede che si produca un documento di specifica per i requisiti dell'ontologia.
- Scenario 2: Riutilizzo e re-ingenierizzazione delle risorse non-ontologiche
- Scenario 3: Riutilizzo delle risorse ontologiche: Search, Compare, Assess, Select.
- Scenario 4: Riutilizzo e re-ingenierizzazione delle risorse ontologiche
- Scenario 5: Riutilizzo e fusione delle risorse ontologiche
- Scenario 6: Riutilizzo, fusione e re-ingenierizzazione delle risorse ontologiche
- Scenario 7: Riutilizzo di ontology design patterns (ODPs)
- Scenario 8: Ristrutturazione di risorse ontologiche
- Scenario 9: Localizzazione di risorse ontologiche

Lo sviluppo di un'ontologia secondo NeOn si sviluppa in un ciclo di vita (*pipeline iterativa*):

1. Fase di inizio;
2. Fase di design;
3. Fase di implementazione;
4. Fase di mantenimento (si ritorna alla fase 2).

Domanda #14

Domanda: Cosa sono i *phased sortals*?

Sono delle entità che rimangono le stesse pur modificando in parte i propri criteri di identità (es. bruco che diventa farfalla). Sono indipendenti (-D), antirigidi (~R) e hanno un criterio di identità (+I).

Domanda #15

Domanda: Quali sono le differenze tra *enduranti* e *perduranti*?

La distinzione chiave tra gli elementi in DOLCE è tra enduranti, cioè le entità temporali e i processi che nel linguaggio naturale sono descritti tramite verbi (correre, mangiare, ecc..) e perduranti, cioè quelle entità che esistono al di fuori dal tempo, cioè gli oggetti ed entità che partecipano ai processi

> *"Gli enduranti sono interamente presenti in ogni istante in cui sono presenti. I perduranti semplicemente si estendono nel tempo accumulando diverse parti temporali in modo che ad ogni istante in cui sono presenti, sono solamente parzialmente presenti."*

Domanda #16

Domanda: In cosa consiste WikiData?

Database collaborativo, basato sui contenuti di Wikipedia e Wikimedia in cui i dati sono strutturati in modo da supportare l'accesso da parte di persone e macchine. È un secondary database: le informazioni sono etichettate con la sorgente da cui provengono. Le reference esterne (a fonti autorevoli) cercano di garantire la massima accuratezza indicando eventualmente anche le fonti degli allineamenti verso altri DB.

Il principio di creazione è opposto a quello di DBpedia: si parte infatti inserendo dati in uno schema strutturato considerato corretto già dall'origine, attraverso un'interfaccia utente predefinita con specifico datamodel. Gli item di wikidata rappresentano concetti astratti o conceti e contengono tutti almeno una label, una descrizione e degli alias.

È bene notare che Wikidata non è un sistema di ragionamento (non ci sono inferenze), bisogna conoscere il vocabolario. (eg. Gli *scienziati nati a Torino* non è un sotto insieme degli *scienziati nati in Italia*).

Domanda $17

Domanda: Quali sono le tipologie di link nelle triple RDF?

Abbiamo principalmente 3 tipologie di links:

- **Relationship links**: punta a oggetti collegati in altre sorgenti di dati.

- **Identity links**: punta uno o più URI che fungono da alias in altre sorgenti per la stessa entità.
- **Vocabulary links**: puntano alle definizioni dei termini del vocabolario.
  - Nei LD, una descrizione è un misto di termini diversi provenienti da vocabolari RDF diversi.
  - È consigliato utilizzare vocabolari standard, ma non sempre è possibile.
    - Bisogna far sì che i propri termini siano de-referenziabili (e collocati in un vocabolario che li descrive).
  - È possibile stabilire una relazione tra termini già esistenti in altri vocabolari.
    - `owl:equivalentClass` e `equivalentProperty`.
    - `rdf:subClassOf` e `rdfs:subPropertyOf` (relazione più lasca).
    - `skos:broadMatch` e `skos:narrowMatch` (relazione più lasca).