ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA GESTIONALE

TESI DI LAUREA

in
Sistemi Informativi

Innovazione nel Semantic Web:
Evoluzione della base di conoscenza semantica YAGO

CANDIDATO:

TERESA ZAMMUTO

RELATORE:

Chiar.ma Prof. WILMA PENZO

Anno Accademico 2015/16

Sessione II

I N D I C E

Introduzione

Lo studio delle basi di conoscenza è il tema principale in cui si colloca questa tesi di laurea.

Tale tema è collocato nell'ambito dei Linked Data, una modalità di pubblicazione di dati strutturati, atti ad essere collegati fra loro al fine di rendere possibili interrogazioni semantiche.
Tim Berners-Lee è il primo a presentare i Linked Data alla conferenza TED del 2009.

"L'idea dei Linked Data è di pensare ogni dato come una singola scatola e cercare di creare il maggior numero di collegamenti tra esse: più collegamenti vengono creati, maggiore sarà il potenziale dei dati posseduti. Ci sono dati in ogni aspetto della vita: se ognuno di noi aggiungesse un bit di conoscenza e tutti questi venissero collegati, si disporrebbe di un'enorme risorsa di conoscenza."[1]

La pubblicazione di Linked Data permette agli sviluppatori di stabilire più facilmente collegamenti tra informazioni provenienti da fonti diverse, rendendo possibili applicazioni nuove ed innovative. Un'applicazione potrebbe, ad esempio, rendere fruibile una vasta mole di dati, al momento non strutturati, combinando i dati di differenti settori: economici, emissioni ambientali, istituti scolastici, etc[2].

---

[1] *http://www.ted.com/talks/tim_berners_lee_on_the_next_web.html*

[2] *https://data.europa.eu/euodp/it/linked-data*

La possibilità figurata dall'ideatore del World Wide Web è ad oggi la nuova sfida nel campo dei database e ha portato, negli ultimi anni, allo sviluppo di solide basi di conoscenza semantiche.

Nel presente elaborato viene fornita un'introduzione preliminare circa le basi di conoscenza: si trattano le necessità che hanno portato al superamento di database tradizionali e come siano strutturate tali innovative ontologie. Un esempio di Knowledge Base è YAGO, su cui si è focalizzato il progetto della tesi. Si tratta di un sistema per la modellazione e l'interrogazione della conoscenza consolidato e maturo del quale viene stata fornita una descrizione complessiva dei vari aspetti caratteristici.
Sono state trattate poi le versioni successive della stessa KB. Si tratta di YAGO2, in cui nuove modalità di estrazione di dati hanno permesso di arricchire la base di conoscenza con informazioni spaziali, e YAGO3, in cui nuovi tool hanno reso possibile l'acquisizione di informazioni estratte da pagine Web redatte in lingue differenti.
L'elaborato si conclude con un capitolo riguardante alcune tematiche di gestione delle basi di conoscenza e ambiti di ricerca futuri.

Capitolo 1 - Basi di conoscenza

1.1 Evoluzione del Web

Il significativo sviluppo di Internet negli ultimi decenni ha reso la rete una delle principali fonti di informazione. Gran parte delle pagine Web sono indicizzate grazie a motori di ricerca capaci di trovare in pochi secondi quelle che contengono la parola chiave richiesta dall'utente. Solitamente l'utente sfoglia alcune pagine Internet tra i risultati trovando rapidamente l'informazione desiderata [4]. Internet si configura così come una grande sorgente di informazioni e possiede il potenziale per diventare la più grande enciclopedia digitale disponibile.
Ciò al momento è solo potenziale poiché Internet raccoglie una grande vastità di contenuti che differiscono per qualità dell'informazione, attendibilità, rilevanza e struttura dei dati [22]. L'utente infatti sperimenta i limiti del Web quando si trova a rivolgere interrogazioni complesse al motore di ricerca. Supponiamo di voler conoscere i titoli dei libri di narrativa pubblicati tra il 1970 e il 1990 il cui autore abbia insegnato all'Università di Bologna. Nessuna delle pagine Web offre una risposta adeguata a questa interrogazione. Se interroghiamo il motore di ricerca con la frase "titoli dei libri di narrativa pubblicati tra il 1970 e il 1990 il cui autore ha insegnato all'Università di Bologna" vengono proposte pagine riguardanti varie case editrici. Lo stesso risultato poco soddisfacente è ottenuto se si tentano altre ricerche con le parole "1970", "1990", "libro", "autore", "docente", "Università di Bologna". Il risultato che si vuole ottenere è un elenco di titoli: nella lista comparirebbe anche "Il nome della rosa",

opera di Umberto Eco dell'anno 1980. Al fine di ottenere l'informazione richiesta, l'utente deve quindi in primo luogo cercare informazioni sui titoli dei libri di narrativa pubblicati tra il 1970 e il 1990, cercare per ognuno di essi l'autore, infine cercare per ogni autore la carriera.
Google infatti è in grado di rispondere efficacemente alla domanda dell'utente solo se vi è la risposta esatta in qualche pagina Web; in caso contrario il motore di ricerca non sarà di grande utilità. Ciò è dovuto al fatto che il Web attualmente possiede file piuttosto che conoscenza [4].
È necessario creare un database la cui struttura permetta di attingere efficacemente ad una vasta quantità di informazioni fra esse correlate. Si è figurata quindi la possibilità di creare una base di conoscenza completa, interpretabile in modo automatico dal calcolatore, riguardo il mondo che ci circonda [22]. Ciò è necessario poiché l'utente interroga la macchina attraverso il linguaggio naturale, ricco di contenuto semantico che attualmente i motori di ricerca non riconoscono. È importante che il computer sia in grado di sollevare l'utente da processi cognitivi onerosi (si pensi all'esempio precedente, in cui l'utente è costretto a cercare i libri pubblicati, controllare gli autori e poi la loro carriera). Grazie a una base di conoscenza con queste caratteristiche sarebbe possibile interrogare il sistema con domande complesse ottenendo direttamente la risposta cercata.

1.2 Modello entità-classe-relazione

Una raccolta di conoscenza come figurata nella sezione 1.1 è chiamata ontologia. Nel più semplice dei casi, un'ontologia è un grafo orientato i cui nodi sono detti *entità* e gli archi sono *relazioni* [4].

La Figura 1 fornisce un esempio di grafo applicato all'esempio sopracitato.

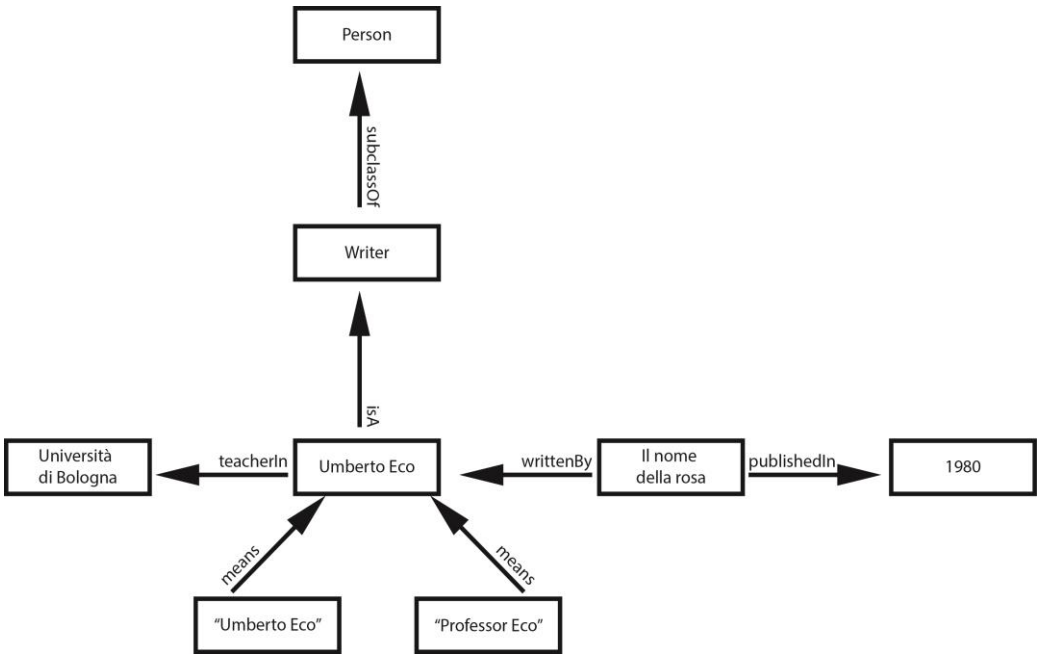

Figura 1: Esempio di grafo.

`Il nome della rosa` e `Umberto Eco` sono due entità collegate dalle relazione `writtenBy` dal momento che quel libro è stato scritto da quell'autore.
Le entità con molti aspetti in comune vengono raccolte a loro volta in *classi*. `Umberto Eco` ed altri autori appartengono alla classe `scrittori`. In questa ontologia la classe `scrittori` è a sua volta una entità alla quale i singoli autori sono collegati tramite una

relazione `isA`. Ogni scrittore è una persona, così che `scrittore` e `persona` sono collegati dalla relazione `subclassOf`.

Tutto questo si conclude con una gerarchia di classi in cui le più generiche (*upper class*) includono quelle più specifiche (*lower class*) [4]. Le classi risultano quindi organizzate in una tassonomia [18].
Un'altra astrazione riguarda la differenziazione tra parola e significato. Si fa distinzione tra "Umberto Eco" (parola) e Umberto Eco (l'autore). Ciò è necessario poiché parole diverse possono riferirsi allo stesso individuo (ad esempio le parole "Umberto Eco", "professor Eco"). Così come la stessa parola può riferirsi a più individui (possono esistere persone omonime del celebre scrittore).
Si astrae anche la scelta della lingua in modo tale che le parole "scrittore", "writer" e "écrivain" si riferiscano a un'unica classe `writer`.
Uno degli assiomi base afferma che un'entità appartiene a tutte le classi superiori ad essa: sapendo che Umberto Eco è uno scrittore e che scrittore è una sottoclasse di persona, deduciamo che Umberto Eco è una persona.
Un sistema di assiomi infine può esprimere che due relazioni siano contrarie, che alcune relazioni ne implichino altre [4]. In questo modo il computer sa che Umberto Eco, avendo pubblicato un libro nel 1980, è sopravvissuto al secondo conflitto mondiale.
La figura 2 rappresenta un esempio di grafo orientato più articolato.

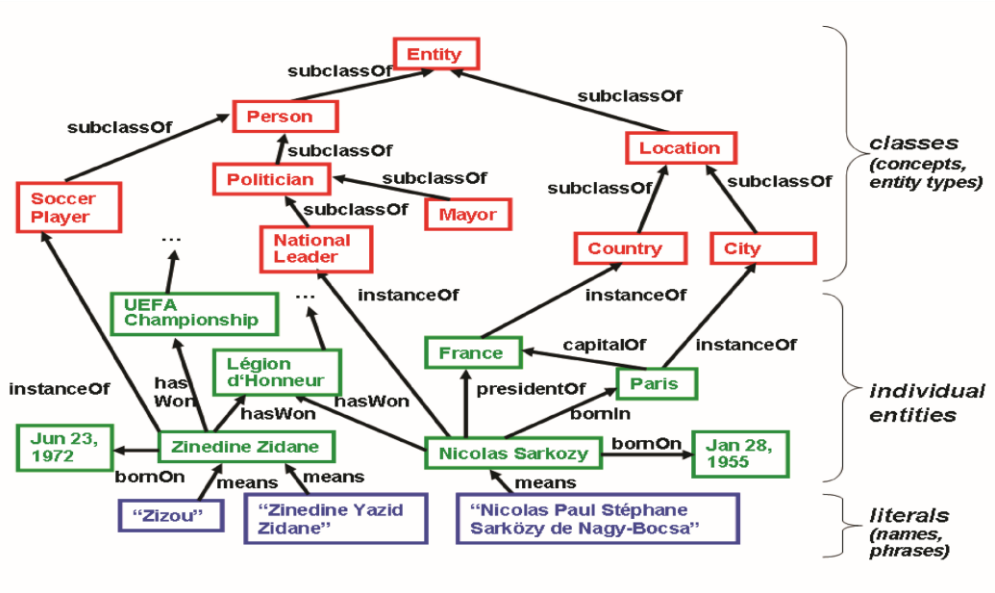

Figura 2: Esempio di grafo orientato [9].

1.3 Fasi di vita di una base di conoscenza

È comunemente accettato che il ciclo di vita per la costruzione di un tale sistema si compone di tre fasi: *knowledge creation*, *knowledge exploitation*, *knowledge maintenance* [11].
Ogni fase è a sua volta composta da tre stadi come mostrato in figura 3.

Knowledge creation	Knowledge exploitation	Knowledge maintenance
Knowledge acquisition	Knowledge reasoning	Knowledge meta-modeling
Knowledge representation	Knowledge retrieval	Knowledge integration
Knowledge storage and manipulation	Knowledge sharing	Knowledge validation

Figura 3: Discipline che compongono il ciclo della conoscenza [11].

1. Knowledge creation

- *Knowledge acquisition.* Il processo di acquisizione automatica inizia estraendo concetti e relazioni tra concetti da testi e documenti usando specifiche metodologie per l'estrazione della terminologia. Allo stesso modo si estraggono istanze concrete dei concetti. Questa fase solitamente prevede l'utilizzo di tecniche per il processamento del linguaggio naturale. Infine tecniche statistiche e simboliche sono usate per estrarre relazioni tra termini e concetti.
- *Knowledge representation.* In questa fase si forniscono specifiche formali sul dominio della conoscenza. Si usano notazioni logiche per rappresentare i concetti e le loro proprietà, le relazioni esistenti tra essi. Tali notazioni logiche usano espressioni e strutture simboliche quali tassonomie, classi e assiomi.
- *Knowledge storage and manipulation.* Si progetta un supporto fisico e logico, sul quale le applicazioni e gli utenti possono fare affidamento per archiviare e condividere la conoscenza. È possibile considerare ogni base di conoscenza come un database avente uno schema, concetti, ruoli e insiemi di istanze. Per questo motivo le tecnologie dei database sono il metodo chiave solitamente usato.

2. Knowledge exploitation

- *Knowledge reasoning.* Consiste nell'inferire conseguenze logiche da un insieme di fatti o assiomi, ottenendo un insieme più ricco su cui lavorare. Attraverso questa fase, infatti, è possibile derivare fatti che non sono espressi esplicitamente nella base di conoscenza.
- *Knowledge retrieval.* Aiuta l'utente o le applicazioni server a trovare la conoscenza di cui hanno bisogno

attraverso interrogazione, navigazione o esplorazione. L'obiettivo è restituire informazioni in una forma strutturata, coerente con i processi cognitivi umani.

- *Knowledge sharing.* Avviene uno scambio di "unità di conoscenza" tra entità così che ogni entità abbia accesso a più informazioni di quante già possieda. Questa fase è svolta al momento in maniera poco efficiente poiché i meccanismi di scambio di informazioni sono relativi solo a domini molto specifici: ciò riduce la propagazione di conoscenza sia in termini di spazio che di tempo. Per risolvere questo problema è stato progettato il linguaggio KIF (Knowledge Interchange Format) [11] che viene usato per scambiare informazioni tra sistemi specifici.

3. Knowledge maintenance

    - *Knowledge meta-modeling.* Questo processo aggiunge descrizioni esplicite riguardo al modo in cui la base di conoscenza deve essere costruita. Comprende in particolare una specifica formalizzazione riguardo le notazioni del dominio, un deposito centralizzato di dati riguardanti formati, origine e relazione tra dati.
    - *Knowledge integration.* Vengono incorporate nuove informazioni in un corpo di conoscenza già esistente grazie a un approccio interdisciplinare. Questo processo determina in che modo le nuove informazioni interagiscano con la conoscenza già costruita e, infine, come le informazioni nuove e vecchie debbano modificarsi per poter essere integrate in un unico corpo. Una delle tecniche utilizzabili è quella del semantic matching. Si riesce così ad andare oltre il significato letterale delle parole per operare a livello del concetto cui le varie espressioni lessicali si riferiscono.

- *Knowledge validation.* Ci si assicura che un elemento sia corretto e conforme agli standard. È infatti necessario assicurarsi che i dati raccolti rientrino nei limiti stabiliti. Scopo di questa fase è infatti sottoporre la base di conoscenza ai test effettuati dagli esperti.

Per quanto riguarda la creazione di una base di conoscenza è opportuno evidenziare un approccio alternativo a quello automatico: inserire entità e relazioni manualmente. La maggior parte delle ontologie odierne sono state infatti compilate manualmente. Si può pensare a WordNet, Sumo, Cyc: sono ontologie che contengono milioni di fatti e assiomi. Ciononostante nessuna di queste fornisce informazioni rilevanti sugli ultimi fatti accaduti. Infatti un'ontologia redatta manualmente non riesce a tenere il passo dell'aggiornamento degli avvenimenti [4]. La compilazione manuale perciò risulta più costosa in termini di tempo e risorse economiche [11].

1.4 Applicazioni

La creazione e lo sviluppo di basi di conoscenza rappresenterebbe una grande risorsa dalle molteplici applicazioni. Tra queste vediamo [22]:

- Un'enciclopedia digitale interpretabile dal computer che può essere interrogata con alta precisione, come un database semantico.
- Un'importante risorsa per la disambiguazione delle entità attraverso una mappatura veloce e accurata di frasi testuali in entità note.

- Un supporto per la ricerca semantica nel Web orientata secondo una struttura “entità-relazione”, così da individuare entità e relazioni nelle pagine Web e ragionare su di esse.
- Una colonna portante per rispondere a interrogazioni espresse con il linguaggio naturale.
- Uno strumento per la traduzione e l’interpretazione del parlato.
- Uno stimolo per l’acquisizione di nuova conoscenza e per il mantenimento automatizzato delle basi di conoscenza acquisite.

Alcuni dei campi che quindi potrebbero beneficiare dai risultati di queste ricerche sono [11]:

- *Il supporto nelle decisioni finanziarie.* L’industria dei servizi finanziari ha da sempre usato sistemi sofisticati. Alcuni sviluppi in questo ambito potrebbero essere importanti per migliorare i sistemi tradizionali aggregando nuove risorse di informazioni, migliorando le performance in tempo reale, spiegando i fondamenti logici che sono dietro le decisioni finanziarie.
- *Sistemi per interrogazioni e risposte.* L’obiettivo di un sistema di interrogazione è quello di fornire conoscenza rilevante per l’utente. In caso di innovazioni si potrebbero costruire dei sistemi di Q&A (Questioning and Answering) molto interessanti. I progressi potrebbero infatti aiutare la diffusione di questi strumenti in tanti campi quali l’istruzione, l’eTourism, la personal finance.
- *Ricerca scientifica.* Gli scienziati hanno bisogno di ottenere facilmente informazioni riguardo composti chimici, sistemi biologici, malfunzionamenti e interazioni tra queste entità. Per questo serve che i dati siano efficacemente integrati per fornire all’utente una visuale completa su una determinata

situazione biologica. I progressi nella costruzione automatica e nel mantenimento di una base di conoscenza aiuterebbero a lavorare meglio con le informazioni necessarie.

Capitolo 2 - YAGO

2.1 Introduzione

YAGO (Yet Another Great Ontology) è una base di conoscenza sviluppata a Saarbrücken, in Germania, da ricercatori del Max Planck Institute for Computer Science nel 2006 [9] e negli anni ha subito continui miglioramenti e estensioni, che continuano tuttora. L'esperienza tedesca in esame combina un'ampia copertura con un'elevata precisione [17] ed è per questo significativa nell'ambito del semantic Web.
YAGO rappresenta tutti i fatti sotto forma di relazioni unarie o binarie: classi di entità e coppie di entità collegate da specifiche relazioni. Il modello di dati può essere visto come un grafo in cui entità e classi corrispondono a nodi e in cui le relazioni corrispondono a archi orientati. La conoscenza è organizzata in formato RDF soggetto-proprietà-oggetto [19]: due nodi adiacenti e l'arco che le connette compongono una tripla [9].
Il linguaggio RDF (Resource Description Framework) descrive i concetti e le relazioni su di essi attraverso l'introduzione di triple (soggetto-predicato-oggetto), e consente la costruzione di query basate su triple pattern, congiunzioni logiche, disgiunzioni logiche e pattern opzionali. RDF è uno strumento base proposto dal World Wide Web Consortium che consente l'elaborazione automatica delle risorse reperibili sul Web. Esso è di fondamentale importanza per la codifica, lo scambio e il riutilizzo di metadati strutturati, e consente l'interoperabilità tra applicazioni che si scambiano informazioni machine-understandable sul Web [13].

La sezione 2.4 di questo capitolo approfondisce il modello di YAGO.
Per creare questa KB sono stati raccolti fatti riguardanti entità dalle category e dalle infobox di Wikipedia, le quali sono state unificate con il dizionario semantico WordNet [19]. Le sorgenti e l'estrazione dell'informazione, elementi caratteristici di YAGO, vengono trattate rispettivamente nelle sezioni 2.2 e 2.3.
La base di conoscenza può essere interrogata tramite SPARQL [19], un linguaggio di interrogazione per dati rappresentati tramite RDF. Nella sezione 2.5 vengono approfondite le modalità di interrogazione della KB.

2.2 Sorgenti di YAGO

Le principali sorgenti di YAGO sono Wikipedia e WordNet.

2.2.1 Wikipedia

Il nucleo di YAGO si basa su Wikipedia, una delle più complete enciclopedie digitali disponibili [17]. Si tratta di una enciclopedia multilingue redatta da volontari e disponibile gratuitamente. Sono 287 le lingue disponibili e la versione inglese possiede 4,5 milioni di articoli [10]. Ogni articolo di Wikipedia è una pagina Web e solitamente descrive un singolo argomento o entità. La particolarità della base di conoscenza YAGO sta nel fatto che le pagine di Wikipedia non vengono sottoposte all'elaborazione del linguaggio naturale: si usa invece un approccio basato sulle sue *infobox* e *category* [17].

Una pagina Wikipedia può avere una infobox. Una infobox è una tabella standardizzata contenente informazioni circa l'entità descritta nell'articolo. Quella relativa alle persone, ad esempio,

contiene la data di nascita, la professione, la nazionalità. Le infobox sono molto più facili da analizzare e sfruttare rispetto ai testi in linguaggio naturale [17].

La maggior parte delle pagine Wikipedia sono state assegnate manualmente a una o più categorie (category). La pagina di Elvis Presley, ad esempio, è nelle categorie “cantanti rock americani”, “nati nel 1935” e in altre 34. Le categorie sono organizzate in maniera gerarchica, ma questa struttura è utile solo parzialmente. Elvis Presley appartiene alla super-category “Grammy Awards” quando in realtà dovrebbe essere un “Grammy Awards winner”. Le informazioni fornite da WordNet risultano a questo punto importanti poiché esso fornisce una gerarchia di concetti lineare e molto precisa [17].

2.2.2 WordNet

WordNet è un dizionario semantico per la lingua inglese sviluppato al Cognitive Science Laboratory dell’università di Princeton [19]. Questo database si propone di organizzare, definire e descrivere i concetti espressi dai vocaboli: distingue tra parole come esse appaiono letteralmente nei testi e il loro significato. Un insieme di parole con lo stesso significato compongono un *synset* (dalla contrazione di synonym set). Ogni synset corrisponde a un significato semantico. Le parole con più di un significato appartengono a più di un synset. La versione WordNet 3.0 contiene 82115 synset e 117798 termini [17]. Cinque relazioni possono avere luogo tra i synset:

- *Iperonimia*: Y è un iperonimo di X se ogni X è "una specie di" Y (es: fiore è iperonimo di rosa);

- *Iponimia*: Y è un iponimo di X se ogni Y è (una specie di) X (es: rosa è iponimo di fiore);
- *Olonimia*: Y è un olonimo di X se X è parte Y (es: automobile è olonimo di volante);
- *Meronimia*: Y è un meronimo di X se Y è parte X (es: volante è meronimo di automobile);
- *Coordinazione*: Y è un termine coordinato di X se X e Y hanno un iperonimo in comune (rosa e viola sono coordinati poiché hanno in comune l'iperonimo fiore).

Nel database sono rappresentati anche verbi, avverbi, aggettivi e relazioni tra essi. WordNet sa che i biologi sono scienziati e che gli scienziati sono esseri umani; tuttavia non riesce a tenere il passo con le nuove entità emergenti (politici attuali, informatici, automobili ibride, etc). Per questo motivo Wikipedia costituisce una risorsa fondamentale per le informazioni di cui dispone [22].

2.3 Estrazione dell'informazione

YAGO inizializza il suo sistema di classi importando tutte quelle di WordNet e le relazioni esistenti tra esse [22].
Dal momento che Wikipedia conosce molte più entità rispetto a WordNet, gli elementi di YAGO vengono estratti da Wikipedia. Ogni titolo delle pagine Wikipedia è unico ed è candidato a diventare un entità nella KB [17]. Ogni entità che YAGO trova in Wikipedia deve essere associata ad almeno una delle classi esistenti nella KB, estratte da WordNet. Se questo tentativo fallisce la suddetta entità e i fatti a essa collegati non vengono inclusi nella base di conoscenza.
In questo caso le category di Wikipedia sono confrontate con i nomi delle classi di WordNet e con i sinonimi delle classi. Viene usato un

decodificatore di nomi per effettuare il confronto, eliminare le category non tassonomiche e stabilire quale sia la parola chiave della category. La parola chiave della categoria `American folk music of the 20th century`, ad esempio, è "music". Se la parola chiave è plurale o può essere estesa a una forma plurale, si cerca una corrispondenza con WordNet tenendo conto anche del termine che qualifica la parola chiave (nell'esempio, "folk music"). Se il risultato del confronto è una identità di nomi, l'entità dapprima esclusa viene aggiunta alla classe di WordNet nella KB; la category responsabile della corrispondenza diviene una lower class della classe nella KB. Questo serve a mantenere coerente la KB [22].
Non tutta la conoscenza è rappresentata in una forma strutturata come descritto fino ad adesso. La maggior parte delle informazioni contenute nei siti Internet è infatti racchiusa nei testi, nascosta nel linguaggio naturale. Per raccogliere questo tipo di conoscenza viene utilizzato l'approccio *Pattern Matching* [4]. Volendo aggiungere una nuova data di nascita all'ontologia, ad esempio, si deve in primo luogo guardare quali sono i siti Web che contengono date di nascita per trovare il modello in cui solitamente viene espressa questa informazione. Lo schema più diffuso con cui si esprime la data di nascita di un individuo in inglese è "X was born on Y" (Umberto Eco was born on January 5th, 1932). Si possono cercare altre istanze di questo pattern in Internet e aggiungerle all'ontologia.
Gli studiosi del Max Planck Institute hanno dovuto ridefinire l'approccio Pattern Matching poiché è risultato talvolta inefficiente [4]. La frase "Umbero Eco, the great writer, was born on 5 January 1932" non rispetta il pattern "X was born on Y"; tuttavia contiene l'informazione cercata. È stato quindi necessario perfezionare l'approccio così da considerare la struttura grammaticale della frase. In questo modo l'unica richiesta è che "X" sia il soggetto del predicato "was born", il quale è connesso a "Y" tramite "on".

Questo approccio è stato implementato in un tool chiamato *Leila* [4]. Per evitare che il software fosse ingannato dall'incertezza del linguaggio naturale e ipotizzasse falsi pattern, è stato effettuato un test statistico sottoponendo a Leila alcuni campioni. Il risultato ha confermato che il tool estrae principalmente fatti corretti. Ad esempio, può apprendere da un insieme di articoli di Wikipedia che il "male di vivere" è un sentimento, che Calcutta si trova sulla foce del Gange e Parigi lungo la Senna [4].

2.4 Il modello di YAGO

Per poter accogliere i dati estratti e per essere pronto ai futuri ampliamenti, YAGO deve essere basato su un sofisticato modello dei dati. Per questo motivo gli sviluppatori della KB hanno introdotto una leggera estensione di RDF: *YAGO model* [17]. Si forniscono due descrizioni: una informale e una formale.

2.4.1 Descrizione informale

YAGO model rappresenta la conoscenza allo stesso modo degli RDF. Tutti gli oggetti (città, persone, URL) sono rappresentati come entità in YAGO. Due entità possono partecipare a una relazione. Per esprimere che Guglielmo Marconi ha vinto il premio Nobel diciamo che l'entità `GuglielmoMarconi` partecipa alla relazione `hasWonPrize` con l'entità `Nobel`.

```
GuglielmoMarconi    hasWonPrize    Nobel
```

Anche numeri, date e altre stringhe possono essere rappresentate come entità.

```
GuglielmoMarconi    bornInYear     1874
```

Le entità sono estratte da oggetti ontologici che sono idealmente indipendenti dal linguaggio. Il linguaggio utilizza parole per riferirsi a tali oggetti. Anche le parole sono entità: parole diverse possono riferirsi alla stessa entità. Allo stesso modo la stessa parola può riferirsi a più entità. Si usano le virgolette per distinguere le parole dalle altre entità.

```
"Guglielmo Marconi" means     GuglielmoMarconi

"Marconi"           means     GuglielmoMarconi
```

Entità simili sono raggruppate in classi. La classe `physicist` comprende tutti i fisici; ogni entità è una *instance* (istanza) di almeno una classe. Questo è espresso dalla relazione `type`.

```
GuglielmoMarconi    type      physicist
```

Le classi stesse sono entità. Per questo motivo ognuna di esse è istanza di un'altra classe. Esse sono perciò organizzate in una gerarchia tassonomica espressa dalla relazione `subClassOf`.

```
physicist      subClassOf          person
```

Le relazioni sono entità e questo permette di rappresentare le proprietà delle relazioni. È possibile esprimere che la relazione `subClassOf` è aciclica transitiva (atr), ossia transitiva e priva di cicli.

```
subClassOf          type      atr
```

La tripla composta da entità, relazione e entità è chiamata *fatto*. Le due entità sono dette *argument* (argomenti) del fatto. In YAGO si

tiene memoria della fonte di ogni fatto. A tal fine ogni fatto ha un *fact identifier,* che è parte integrante del YAGO model, a differenza di quanto avviene in RDF. Supponiamo che al fatto (`GuglielmoMarconi`, `bornInYear`, `1874`) sia assegnato il fact identifier `#1`. Allora è possibile affermare:

```
#1        foundIn   Wikipedia
```

Vengono chiamate *common entity* (entità comuni) tutte le entità che non si riferiscono a fatti o relazioni. Le common entity che non sono classi vengono chiamate *individual* (individui) [17].

2.4.2 Descrizione formale

YAGO si presenta quindi come una funzione che associa fact identifier a triple. Più formalmente, questa KB può essere descritta come un *reification graph* (grafo di reificazione).
Un generico grafo di reificazione è definito su:

- Un insieme di *node* (nodi) N. Questi sono le common entity.
- Un insieme di *edge identifier* (identificatori di archi) I. Questi sono i fact identifier.
- Un insieme di *label* (etichette) L. Queste sono i nomi delle relazioni.

Il reification graph è una funzione iniettiva totale:

$G_{N,I,L} : I \rightarrow (N \cup I) \times L \times (N \cup I)$

L'immagine di questa funzione è costituita dagli spigoli del grafo. Ogni spigolo è unico e ha un identificatore in I. Gli spigoli possono

connettere non solo due nodi ma anche un nodo e uno spigolo oppure due spigoli. Ogni spigolo ha anche un'etichetta da L. L'ontologia YAGO basata su un insieme finito di entità C, un insieme finito di nomi di relazioni R e un insieme finito di fact identifier I è un grafo di reificazione. La funzione, iniettiva totale, si esprime:

$Y : I \rightarrow (I \cup C \cup R) \times R \times (I \cup C \cup R)$

YAGO viene redatta elencando gli elementi della funzione nella forma:

id_1 : $arg1_1$ rel_1 $arg2_1$
id_2 : $arg1_2$ rel_2 $arg2_2$
…

Dove id_1 è il fact identifier #1, rel_1 è la relazione di tale fatto e $arg1_1$ e $arg2_1$ ne sono rispettivamente il primo e il secondo argomento.
Si consideri ora la seguente coppia di fatti:

id_1 : $arg1_1$ rel_1 $arg2_1$
id_2 : id_1 rel_2 $arg2_2$

Qualora il fact identifier id_1 non compaia in altri fatti oltre il id_2, è ammessa una forma abbreviata e il fact identifier id_1 può essere omesso:

id_2 : ($arg1_1$ rel_1 $arg2_1$) rel_2 $arg2_2$

ciò potrebbe significare, ad esempio:

```
#1:    GuglielmoMarconi  BornInYear     1874
                FoundIn   Wikipedia
```

Infine alcuni fatti richiedono più di due argomenti. Supponiamo di dover esprimere il fatto: Marconi ha vinto il premio Nobel nel 1909. Un approccio comune per trattare questo esempio è l'utilizzo di una relazione n-aria. Nell'esempio citato: `wonPrizeInYear(GuglielmoMarconi, Nobel, 1909)`. Tuttavia il linguaggio RDF non permette relazioni di questo tipo: è possibile invece rappresentare una nuova relazione binaria per ogni argomento.

```
Nobel               prize          MarconiGetsNobel
GuglielmoMarconi    winner         MarconiGetsNobel
1909                year           MarconiGetsNobel
```

In questo modo un fatto n-ario può essere rappresentato da un *event entity* (entità-evento) del tipo (`say`, `MarconiGetsNobel`). YAGO model offre però una soluzione migliore a questo problema. In ogni relazione n-aria si identifica una *primary pair* (coppia primaria) di argomenti: nell'esempio precedente consideriamo la persona e il premio come coppia primaria. Si rappresenta la primary pair come un fatto binario con un fact identifier:

```
#1:  GuglielmoMarconi    hasWonPrize    Nobel
```

Tutti gli altri argomenti possono essere rappresentati come relazioni tra il fact identifier della primary pair e gli altri argomenti:

```
#2:  #1   inYear         1909
```

Usando una sintassi semplificata come precedentemente questo può essere espresso:

```
GuglielmoMarconi        hasWonPrize        Nobel
                inYear        1909
```

2.5 Interrogazione

Per interrogare YAGO è stato progettato un linguaggio che si basa sui concetti di SPARQL [9]. Il sistema NAGA (Not Another Google Answer) implementa tale linguaggio e fornisce una classificazione statistica dei risultati. Una query è un insieme di *fact template*, dove ogni template deve essere associato a uno spigolo e ai nodi su cui esso incide nel grafo della conoscenza [9]. Si consideri la domanda: "Chi è nato dopo Guglielmo Marconi?". La query può essere espressa come segue:

```
Guglielmo Marconi bornOnDate ?e
        ?x bornOnDate ?y
          ?y after ?e
```

NAGA per prima cosa normalizza le notazioni stenografiche in notazioni standard, così che ogni riga della query consista in: fact identifier, primo argomento, relazione, secondo argomento.

```
?i1: "Guglielmo Marconi" bornOnDate ?e
        ?i2: ?x bornOnDate ?y
          ?i3: ?y after ?e
```

Il sistema considera quindi ogni significato possibile delle parole. Questa fase è necessaria poiché le entità possono avere più di un nome. A tal fine gli argomenti che non sono variabili vengono sostituiti da una nuova variabile e viene aggiunto un nuovo fatto "`means`" per essa. Questa operazione viene detta *word resolution* [17]:

```
?i0: “Guglielmo Marconi” means ?GuglielmoMarconi
      ?i1: ?GuglielmoMarconi bornOnDate ?e
             ?i2: ?x bornOnDate ?y
               ?i3: ?y after ?e
```

La figura 4 fornisce una rappresentazione dell’esempio. Una risposta a questa query dovrebbe vincolare le variabili della query originale (`?e`, `?x`, `?y`) a quelle introdotte nella word resolution (`?GuglielmoMarconi`). Inizialmente si eliminano le righe contenenti *filter relation* (relazioni filtro), ossia relazioni che esprimono condizioni che i risultati devono soddisfare. Nell’esempio, l’ultima riga viene temporaneamente eliminata. Quindi viene implementata una unica query SQL contenente un argomento del `SELECT` per ogni variabile che vogliamo vincolare e una join per ogni riga della query. Applicando all’esempio:

```
SELECT f0.arg2, f1.arg2, f2.arg1, f2.arg2
    FROM facts f0, facts f1, facts f2
  WHERE f0.arg1=‘ “Guglielmo Marconi” ’
         AND f0.relation=‘means’
           AND f1.arg1=f0.arg2
       AND f1.relation=‘bornOnDate’
       AND f2.relation=‘bornOnDate’
```

Tale query fornisce valori per le variabili `?GugliemoMarconi`, `?e`, `?x`, `?y`. Successivamente il sistema filtra i risultati sulla base delle condizioni della filter relation: la relazione `after` tra le coppie `?y`, `?e`. Se esistono valori per cui tale relazione è verificata, essi vengono restituiti come risultato.

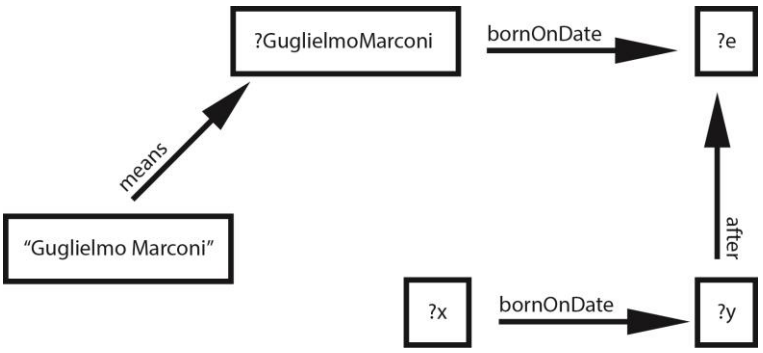

Figura 4: Esempio di query.

2.6 Valutazione generale

In questa sezione la base di conoscenza YAGO viene valutata secondo due parametri frequentemente utilizzati in IR o *information retrieval*, l'insieme delle tecniche per gestire la rappresentazione, la memorizzazione, l'organizzazione e l'accesso ad oggetti contenenti informazioni.
Il primo parametro utilizzato è la *dimensione* di YAGO, che indica la numerosità degli elementi contenuti nella KB, siano essi classi, entità, fatti o relazioni.
Il secondo è la *precisione* della base di conoscenza, che è data dal rapporto tra il numero degli elementi rilevanti contenuti nella KB e il numero degli elementi totali contenuti dalla stessa.

P = (numero di elementi rilevanti recuperati) / (numero di elementi recuperati)

Esistono molti parametri utilizzabili; vengono qui di seguito date le definizioni di *recall* e *misura F*, utilizzati nei capitoli successivi[3].

[3] *https://it.wikipedia.org/wiki/Information_retrieval*

Recall è la proporzione fra il numero di elementi rilevanti recuperati e il numero di tutti gli elementi rilevanti recuperabili:

R = (numero di elementi rilevanti recuperati) / (numero di elementi rilevanti)

Misura F è la media armonica fra precisione e recall

$$F = 2 * P * R / (P + R)$$

2.6.1 Dimensione

Il numero totale di entità contenute è 1,7 milioni, mentre i fatti che si conoscono riguardo esse sono 15 milioni [17]. Le figure 5 e 6 forniscono dati significativi a riguardo. Non è facile confrontare tali valori con quelli di altre ontologie poiché le differiscono per struttura, assiomi, relazioni e qualità. Per questo motivo la figura 7, contente dati sulle altre basi di conoscenza, viene fornita solo a scopo informativo.

Relations		92
Classes		224391
Individuals		1531588
	(People)	546308
	(Locations)	230988
	(Companies)	57893
	(Movies)	33234

Figura 5: Numero di entità di YAGO [17].

Relation	# Facts	Relation	# Facts
hasUTCOffset	12724	hasWonPrize	13645
livesIn	15185	writtenInYear	16441
originatesFrom	16876	directed	18633
hasPredecessor	19154	actedIn	22249
hasDuration	23652	bornInLocation	24400
hasImdb	24659	hasArea	26781
hasProductionLanguage	27840	produced	30519
hasPopulation	30731	isOfGenre	33898
hasSuccessor	46658	establishedOnDate	69529
hasWebsite	79779	created	83627
locatedIn	125738	diedOnDate	168037
subClassOf	211979	bornOnDate	350613
givenNameOf	464816	familyNameOf	466969
inLanguage	2389627	isCalled	2984362
type	3957223	means	4014819

Figura 6: Principali relazioni di YAGO [17].

Ontology	# Entities	# Facts
SUMO	20000	60000
Ponzetto et al.	n/a	110000
WordNet	117659	821492
Cyc	300000	3000000
TextRunner	n/a	7800000
YAGO	1700000	15000000
DBpedia	1950000	103000000

Figura 7: Dimensione di alcune ontologie [17].

2.6.2 Precisione

Per valutare la precisione di un'ontologia, i fatti contenuti devono essere confrontati con la realtà. Tale valutazione di YAGO è avvenuta manualmente: alcuni fatti estratti casualmente dalla base di

conoscenza sono stati presentati a una giuria che ne ha stimato la correttezza [17]. Per ogni fatto i giudici potevano affermare "corretto", "errato" oppure "non conosco". Dato che non è sufficiente il buon senso a valutare la correttezza di YAGO, i giudici potevano usufruire anche delle corrispondenti pagine Wikipedia. La giuria, composta di dodici membri, ha valutato un totale di 5200 fatti. La prova ha avuto esito positivo: la precisione media dei fatti è oltre il 95% [17]. La figura 8 mostra, per alcune delle relazioni, il numero di istanze valutate e la precisione risultante.

	Relation	#Evaluated	Accurancy
1	hasExpenses	46	100.0% ± 0.0 %
2	hasInflation	25	100.0% ± 0.0 %
3	hasLaborForce	43	97.67441% ± 0.0 %
4	during	232	97.48950% ± 1.838 %
5	ConceptualCategory	59	96.94342% ± 3.056 %
6	participatedIn	59	96.94342% ± 3.056 %
7	plays	59	96.94342% ± 3.056 %
8	establishedInYear	57	96.84294% ± 3.157 %
9	createdOn	57	96.84294% ± 3.157 %
10	originatesFrom ...	57	96.84294% ± 3.157 %
	WordNetLinker		
72	...	56	95.11911% ± 4.564 %
74	InfoboxType	76	95.08927% ± 4.186 %
75	hasSuccessor ...	53	94.86150% ± 4.804 %
88	hasGDPPPP	75	91.22189% ± 5.897 %
89	hasGini	62	91.00750% ± 6.455 %
90	discovered	84	90.98286% ± 5.702 %

Figura 8: Precisione delle relazioni di YAGO [17].

Capitolo 3 - YAGO2

YAGO2 è la seconda versione di YAGO sviluppata con l'obiettivo di integrare i fatti della KB con le dimensioni spaziale e temporale. In tale versione sono state introdotte nuove regole dichiarative che stanno alla base della nuova architettura. È stato sviluppato inoltre il nuovo modello di rappresentazione SPOTL(X) per la navigazione e interrogazione di YAGO2.

3.1 Regole dichiarative

L'architettura di YAGO2 si basa su alcune regole dichiarative che sono archiviate in file di testo. Tali regole prendono la forma di triple soggetto-predicato-oggetto (SPO) così da risultare semplici fatti addizionali [7].

- *Factual rule*. Sono la traduzione in forma dichiarativa dei fatti e delle eccezioni inseriti manualmente in YAGO. Sono incluse definizioni delle relazioni, domini, definizioni delle classi che compongono la gerarchia di YAGO2. Queste regole inoltre accrescono la lista delle eccezioni per associare le category di Wikipedia ai synset di WordNet [7]. La category "capital", ad esempio, include importanti città mentre WordNet propone come significato primario un ammontare finanziario. Una factual rule può stabilire di associare la category a un significato secondario del synset, ad esempio con il fatto:

```
"capital" hasPreferredMeaning
  wordnet_capital_108518505
```

- *Implication rule*. Queste regole dispongono che, qualora determinati fatti appaiano nella KB, un altro fatto debba essere aggiunto. In questo modo è possibile dedurre nuova conoscenza da quella già posseduta. Questa regola è rappresentata come una tripla in cui il soggetto è costituito dalle premesse e l'oggetto dalla conclusione. Se la relazione A è una sotto-proprietà della relazione B, ad esempio, allora le istanze della prima sono anche istanze della seconda [7]. Per esprimere ciò viene utilizzata la relazione *implies*:

```
"$1  $2  $3;  $2  subpropertyOf  $4;"
      implies        "$1 $4 $3"
```

- *Replacement rule*. Se parte del testo della sorgente corrisponde a una espressione regolare, tale porzione di testo deve essere sostituita con una determinata stringa. Questa regola permette inoltre di eliminare gli articoli di Wikipedia che non devono essere processati sostituendo il titolo con una stringa vuota. La replacement rule utilizza la relazione *replace* e argomenti composti di stringhe:

```
"\{\{USA\}\}"  replace   "[[United States]]"
```

- *Extraction rule*. Una sequenza di fatti viene creata qualora una sezione del testo corrisponda a un'espressione specifica. Questa regola è applicata principalmente ai pattern delle infobox di Wikipedia, in seguito anche alle category e ai titoli degli articoli.

```
"\[\[Category: (.+) births\]\]"   pattern
                  "$0 wasBornOnDate Date($1)"
```

L’espressione specifica di YAGO per indicare la data di nascita di un soggetto è `x wasBornOnDate y`. Esistendo un formato standard è possibile individuare quali sono le entità da estrarre dalla categoria `births`, quale ruolo semantico abbiano e in che forma inserirle in YAGO. Si estraggono gli elementi della categoria `births`: YAGO crea quindi un fatto in cui il primo argomento equivale al soggetto nella category (o al titolo dell’articolo), il secondo argomento deve essere in formato data ed equivale alla data contenuta nella category, la relazione che collega i due argomenti è `wasBornOnDate`.

Tale architettura per le regole di estrazione è versatile e facilmente estendibile. Nella implementazione attuale le regole di estrazione coprono 200 pattern delle infobox, 90 delle categorie e alcuni pattern relativi alle pagine di disambiguazione. In totale i pattern coprono circa 100 relazioni [7].

3.2 Dimensione temporale

YAGO2 contiene un nuovo tipo di dato, *yagoDate*, che denota istanti temporali. Le date sono indicate nel formato standard `yyyy-mm-dd`. Se l'anno costituisce l'unica informazione conosciuta il formato usato è `yyyy-##-##`. I fatti possono essere collegati soltanto a istanti: un lasso temporale è rappresentato da due relazioni tra un argomento costante e due yagoDate, che insieme compongono un intervallo (ad esempio `wasBornOnDate` e `diedOnDate`). È impossibile ad esempio esprimere che Elvis Presley sia vissuto per 42 anni con un unico fatto il cui secondo argomento sia un yagoDate. È necessario affermare che Elvis nacque nel 1935 e morì nel 1977: questi due istanti identificano per quanto tempo è esistito l’artista.

L'informazione temporale viene mantenuta sia per le entità che per i fatti.

3.2.1 Entità

L'esistenza di molte entità è caratterizzata da un inizio e da una fine. L'intervallo di tempo che intercorre tra questi due istanti è racchiuso nel concetto di *existence time*. Elvis Presley, ad esempio, è associato agli istanti `1953-01-08` come data di nascita e `1977-08-16` come data del decesso.
Esistono poi entità che iniziano a esistere ma non cessano (si pensi ai brani musicali), altre di cui non sono definite inizio e fine (figure mitologiche).

In YAGO2 vengono rappresentate quattro tipologie di collocazione di entità nel tempo. Queste modalità riescono a coprire quasi tutti i casi in cui esiste un a relazione tra entità e tempo [7].

- *Persone*: le relazioni `wasBornOnDate` e `diedOnDate` demarcano il loro existence time.
- *Gruppi*: le relazioni `wasCreatedOnDate` e `wasDestroyedOnDate` indicano l'existence time di club sportivi, università, gruppi musicali e altre organizzazioni.
- *Manufatti*: si tratta di dipinti, libri, canzoni, edifici, per i quali vengono utilizzate le relazioni `wasCreatedOnDate` e `wasDestroyedOnDate`.
- *Eventi*: si pensi a Olimpiadi, campionati di calcio ma anche a epoche storiche con una precisa denominazione. Per queste entità vengono usate le relazioni `startedOnDate` e `endedOnDate`.

Esistono due relazioni entità-tempo che permettono di gestire tutti i casi sopraelencati: `startsExistingOnDate` e `endsExistingOnDate`. Entrambe sono istanze della più generale yagorelation e collegano un'entità e un'istanza di yagoDate.
Altre relazioni più specifiche sono sotto-proprietà delle due principali:

```
wasBornOnDate  subpropertyOf  startsExistingOnDate.
```

Per gli eventi che durano una sola giornata viene usata la relazione `happenedOnDate` che è sotto-proprietà di entrambe le relazioni generali di inizio e fine esistenza.

3.2.2 Fatti

I fatti, allo stesso modo delle entità, possono avere una dimensione temporale. Il fatto `BobDylan created BlondeOnBlonde` è associato alla data in cui l'album è stato pubblicato (`1966-05-16`). Eventi come questo sono caratterizzati da una unica data. Quando vengono estratte informazioni temporali sui fatti si associa a ognuno di questi un *occurrence time*, ossia l'intervallo di tempo in cui il fatto è avvenuto. Un fact identifier e un'istanza di yagoDate possono partecipare a due relazioni: `occursSince` e `occursUntil`. Similmente a quanto avviene per le entità, queste due relazioni indicano genericamente l'inizio è la fine dell'accadimento di un fatto.
I fatti che durano una sola giornata partecipano alla relazione `occursOnDate` con la data interessata. Tale relazione è sotto-proprietà di `occursSince` e di `occursUntil`.

Se infine uno stesso fatto è avvenuto molteplici volte YAGO2 lo contiene più volte con diversi fatti identifier. Si pensi a un cantante che riceve più volte lo stesso premio:

```
#1:  BobDylan  hasWonPrize    GrammyAward

#2:  BobDylan  hasWonPrize    GrammyAward

          #1 occursOnDate 1973

          #2 occursOnDate 1979
```

3.3 Dimensione spaziale

Tutti gli oggetti fisici hanno una posizione nello spazio. YAGO2 considera le entità con una dimensione spaziale permanente, come paesi, città, fiumi. Nella precedente versione di YAGO, così come su WordNet, tali entità non hanno una super-classe comune. È stata introdotta quindi la nuova classe yagoGeoEntity che raggruppa le entità con una posizione permanente sulla terra. Le sue sottoclassi, i cui nomi sono estratti da WordNet, sono ad esempio posizioni, corpi d'acqua, strutture, edifici, strade. La posizione di una geo-entity è descritta dalle sue coordinate geografiche. Il tipo di dato introdotto per archiviare tali informazioni è yagoGeoCoordinates che consiste in coppie di latitudine-longitudine Ogni istanza geo-entity è connessa alla sua posizione tramite la relazione `hasGeoCoordinates` [7].

3.3.1 Entità

YAGO2 fornisce la dimensione spaziale di tre tipi di entità:

- *Eventi*: le battaglie o le competizioni sportive sono collegate alla posizione in cui hanno avuto luogo tramite la relazione `happenedIn`.
- *Gruppi*: la sede di un'organizzazione è espressa tramite la relazione `isLocatedIn`.
- Manufatti: la loro posizione, se permanente, è indicata tramite la relazione `isLocatedIn`.

La semantica di queste relazioni può essere differente ma, piuttosto che trattare separatamente ogni caso, è stata definita una unica relazione: *placedIn*. `isLocatedIn` e `happenedIn` sono sotto-proprietà della più generica `placedIn`.

3.3.2 Fatti

Qualora un fatto sia caratterizzato da una certa dimensione spaziale, il suo fact identifier è connesso alla rispettiva posizione tramite la relazione `occursIn`.

Vengono distinti tre casi in si può dedurre il luogo di un fatto, per ognuno dei quali viene fornita descrizione, regola e esempio. La regola sfrutta una o più implication rule così strutturate: premesse, linea grafica ad indicare "`implies`", tripla derivata dalla implication rule.

- *Permanent relation*. Le relazioni permanenti indicano una associazione diretta per l'entità (`2006FIFAWorldCup isCalled FootballWorldCup2006`). Esiste poi un fatto a indicare se tale entità abbia anche una posizione permanente (`2006FIFAWorldCup happenedIn Germany`). Vengono usate due implication rule al fine di trasferire la posizione dell'entità al fatto (1) e la entità alla sua geo-entity (2):

1: $id: $s $p $o;

$p type permanentRelation;

$s placedIn $1

$id occursIn $1

2: $id: $s $p $o;

$p type permanentRelation;

$s type yagoGeoEntity

$id occursIn $

Esempio:

1: #1: lagoDiGarda isCalled Benaco;

isCalled type permanentRelation;

lagoDiGarda placedIn Italy

#1 occursIn Italy

2: #1: lagoDiGarda isCalled Benaco;

isCalled type permanentRelation;

lagoDiGarda type yagoGeoEntity

#1 occursIn lagoDiGarda

- *Space-bound relation*. Alcuni fatti accadono nel luogo indicato dagli argomenti. Si considerino i due esempi:

```
#1:  BobDylan        wasBornIn      Duluth

#2:  ParisDistrict   hasMayor       AnneHidalgo
```

Il secondo argomento del fatto `#1` racchiude la posizione del fatto stesso. La relazione `wasBornIn` è istanza della classe `relationLocatedByObject`, sottoclasse di `yagoRelation`; `hasMayor` è istanza della classe `relationLocatedBySubject`. È possibile quindi trasferire la posizione di uno degli argomenti al fatto stesso attraverso due regole:

```
1: $id:  $s  $p  $o;

$p type relationLocatedByObject;

$o type yagoGeoEntity
-----------------------
$id occursIn $o
```

```
2: $id:  $s  $p  $o;

$p type relationLocatedByObject;

$o placedIn $1
-----------------------
$id occursIn $1
```

Esempio:

```
1: #1:    BobDylan       wasBornIn      Duluth;

        wasBornIn type relationLocatedByObject;

              Duluth type yagoGeoEntity

                 #1 occursIn Duluth
```

```
2: #2:    AnneHidalgo    isMayorOf      ParisDistrict;

        isMayorOf type relationLocatedByObject;

            ParisDistrict   placedIn Paris

                 #2 occursIn Paris
```

- *Tandem relation.* Alcune relazioni sono concatenate. La relazione `wasBornOnDate`, ad esempio, definisce la dimensione temporale del corrispondente fatto `wasBornIn`. Casi come questo vengono gestiti tramite la relazione `timeToLocation` la quale sussiste tra due relazioni. La prima relazione specifica la data dell'evento e la seconda il luogo. La seguente regola può trasferire l'informazione spaziale da una relazione all'altra:

```
$id1:  $s  $p  $t;

$p timeToLocation $r;

$id2:  $s  $r  $1;
```

```
$id2 occursIn $1
-----------------
$id1 occursIn $1
```

Esempio:

```
#1: BobDylan   wasBornOnDate  1941;

wasBornOnDate timeToLocation wasBornIn;

#2:  BobDylan wasBornIn Duluth;

#2 occursIn Duluth
-----------------
#1 occursIn Duluth
```

3.4 Rappresentazione SPOTL(X)

Si ipotizzi di voler sapere quali sono i concerti che hanno avuto luogo vicino a San Francisco. È necessaria una query piuttosto complessa per ottenere questa informazione:

```
?id:  ?s  performed  ?o .

?id occursIn ?1 .

?1 hasGeoCoordinates ?g .

SanFrancisco  hasGeoCoordinates  ?sf  .

?g  near  ?sf  .
```

Per un utente non esperto risulta difficile comporre queste cinque join. Invece che vedere solo triple SPO, l'utente dovrebbe essere in

grado di vedere tuple estese con le componenti spaziale e temporale. In YAGO2 questo è possibile organizzando le cinque informazioni interessate in quintuple *SPOTL* di dati [7] dove il fatto è già associato alle relative informazioni spaziali e temporali. Esiste una ulteriore estensione in sestuple SPOTLX, in cui l'ultima componente fornisce parole o frasi chiave riguardo il contesto del fatto. Questo ultimo fattore soddisfa quei casi in cui l'utente abbia una buona intuizione riguardo l'informazione richiesta, ma abbia problemi a strutturarla sotto forma di triple SPO.
Il modello di rappresentazione SPOTL(X), introdotto da YAGO2, permette di navigare e interrogare più efficientemente la KB.
Una view SPOTL(X) è composta dalle seguenti relazioni virtuali:

- *R(Id, S, P, O):* quadruple composte da fact identifier, relazione, due argomenti.

- *T(Id, TB, TE):* triple che associano un intervallo di tempo [TB, TE] con il fact identifier Id. La componente TB è impostata usando la relazione `occursSince` mentre TE usando `occursuntil`.

- *L(Id, LAT, LON):* triple che associano la posizione <LAT, LON> al fact identifier Id. Questo fattore è impostato con la relazione `occursin` per trovare il luogo e `hasGeoCoordinates` per le coordinate.

- *X(Id, C):* coppie che associano un contesto C a un fact identifier Id. Tale componente si basa sulla relazione `hasContext`, applicata a soggetto e oggetto del fatto.

Una view SPOTL(X) è definita [7]

$$\pi_{[R.Id,\,[TB,TE],\,<LAT,LON>,\,C]}\Big(\big((R \bowtie_{[Id=Id]} T) \bowtie_{[Id=Id]} L\big) \bowtie_{[Id=Id]} X\Big)$$

effettuando una join tra i fatti di R e le relative informazioni di T, L e X. Viene utilizzata l'outer join per evitare di perdere quelle triple che mancano di informazioni spaziali, temporali o contestuali. La figura 9 mostra un esempio di view SPOTL(X) ispirato all'esempio citato in questa sezione: Grateful dead si esibisce a San Francisco nel 1978 con il brano "The Closing of Winterland".

Id	S	P	O	T	L	X
Id1	GD	performed	TCOW	1978-12-31	-37.5, 122.3	"Wall of Sound…"
Id2	Id1	occursIn	SF			"Golden Gate cowboys…"
Id3	Id1	occursOnDate	1978-12-31			

Figura 9: Esempio di view SPOTL(X) [7].

3.5 Interrogazione

Questa sezione tratta l'interrogazione di YAGO2, che in particolare avviene tramite la *SPOTL(X) query interface*, programmata per operare direttamente con view SPOTL(X). Per trattare le dimensioni di tempo, spazio e contesto sono stati introdotti i predicati mostrati in figura 10.

Dimension	Predicate	Examples
Time	overlaps	[1967,1994][1979,2010]
	during	[1967,1994][1915,2009]
	before	[1967,1994][2000,2008]
	after	[1967,1994][1939,1945]
Space	westOf	<48.52,2.20><52.31,13.24>
	northOf	<48.52,2.20><41.54,12.29>
	eastOf	<48.52,2.20><51.30,0.70>
	southOf	<48.52,2.20><59.20,18.30>
	nearby	<48.52,2.20><48.48,2.80>25.00
conteXt	matches	"... cowboys in Mexico ..."(+ cowboys) "... her debut album ..."(+debut - live)

Figura 10: Predicati introdotti in YAGO2 [7].

Nelle query è possibile aggiungere un predicato per ogni dimensione. Le due scritture sottostanti sono equivalenti e restituiscono le canzoni scritte da George Harrison dopo la morte di John Lennon

```
1:  GeorgeHarrison created ?s .
     ?s wasCreatedOn ?t1 .
    JohnLennon diedOn ?t2 .
        ?t1 after ?t2 .
```

```
2:  GeorgeHarrison created ?s after JohnLennon
```

Attraverso i predicati indicati è possibile ottenere i nomi dei chitarristi ambidestri nati nelle vicinanze di Seattle (entro un raggio di 25 km):

```
?p isA Guitarist matches (+left +handed) .
  ?p wasBornIn ?c nearby Seattle 25.0 .
```

3.6 Valutazione generale

3.6.1 Dimensione

YAGO2 contiene un vasto numero di fatti provenienti da Wikipedia. La figura 11 fornisce dati circa le entità contenute in YAGO2.

Class	#Entities	% existence time	% existence location
People	872155	80.46	–
Groups	316699	38.46	24.03
Artifacts	212003	58.91	1.78
Events	187392	60.16	16.01
Locations	687414	13.34	100
Other	372724	24.99	2.25
Total	2648387	47.05	30.62

Figura 11: Entità contenute in YAGO2 e relative informazioni spazio-temporali [7].

3.6.2 Precisione

La valutazione della precisione di YAGO2 riguarda i fatti estratti da Wikipedia, trascurando invece quelli derivanti dalle implication rule. Fatti selezionati casualmente sono stati sottoposti a una giuria responsabile di giudicarne la correttezza, la quale poteva servirsi di

pagine Wikipedia qualora necessario. I 26 giudici partecipanti hanno vagliato 5864 fatti. La valutazione ha mostrato un'elevata accuracy: il 97,80% dei fatti era esatto. La precisione media è risultata del 95,40%.
Si forniscono tabelle significative che indicano: Valutazione di relazioni non-spaziali e non temporali (figura 12), valutazione di relazioni temporali e spaziali (figura 13), esempi delle relazioni più e meno accurate (figura 14).

Relation	#Total facts	#Evaluated	Accuracy
actedIn	126636	69	97.36% ± 2.64 %
created	225563	94	98.04% ± 1.96 %
exports	522	113	93.22% ± 4.32 %
graduatedFrom	15583	57	96.84% ± 3.16 %
hasExport	161	61	95.50% ± 4.21 %
hasGender	804747	50	94.58% ± 5.07 %
hasGivenName	746492	134	97.16% ± 2.43 %
hasLatitude	311481	47	96.22% ± 3.78 %
holdsPoliticalPosition	3550	81	94.20% ± 4.53 %
influences	18653	58	95.28% ± 4.42 %
isInterestedIn	296	93	92.85% ± 4.83 %
isMarriedTo	27708	58	96.89% ± 3.11 %
subclassOf	367040	339	93.42% ± 2.67 %
type	8414398	208	97.68% 1.83 %

Figura 12: Valutazione di relazioni non-spaziali e non temporali [7].

Relation	#Total facts	#Evaluated	Accuracy
diedIn	28834	88	97.91% ± 2.09 %
diedOnDate	315659	79	97.68% ± 2.32 %
happenedIn	11694	51	96.50% ± 3.50 %
happenedOnDate	27563	94	97.86% ± 2.14 %
isLocatedIn	436184	51	96.50% ± 3.50 %
livesIn	20882	56	96.79% ± 3.21 %
wasBornIn	90181	49	96.36% ± 3.64 %
wasBornOnDate	686053	56	96.79% ± 3.21 %
wascreatedOnDate	507733	110	97.43% ± 2.41 %
wasDestoyedOnDate	23617	72	96.15% ±3.61 %

Figura 13: Valutazione di relazioni spaziali e temporali [7].

Relation	#Total facts	#Evaluated	Accuracy
created	225563	94	98.04% ± 1.96 %
diedIn	28834	88	97.91% ± 2.09 %
happenedOnDate	27563	94	97.86% ± 2.14 %
	.		
	.		
	.		
hasHeight	26477 547	120	91.99% ± 4.59 %
hasBudget	574	95	90.97% ± 5.41 %
hasGDP	175	93	90.79% ± 5.52 %

Figura 14: Accuratezza di alcune relazioni [7].

Capitolo 4 - YAGO3

YAGO3 è l'ultima estensione di YAGO e unifica informazioni provenienti dalle diverse edizioni linguistiche di Wikipedia. Le informazioni sono state estratte da 10 edizioni di Wikipedia: inglese, tedesca, francese, danese, italiana, spagnola, romena, polacca, araba e farsi [10].

4.1 La creazione di una KB da Wikipedia multilingue

L'ampliamento della KB è stato raggiunto tramite il conseguimento di tre sfide [10] descritte in questo paragrafo.

4.1.1 Creazione di un insieme di entità

Una stessa entità in Wikipedia può essere trattata in diverse lingue. Gli sviluppatori di YAGO3 hanno dovuto quindi porre attenzione alla fase di estrazione delle entità al fine di non creare duplicati. A tale scopo si sono serviti di Wikidata, una KB basata su crowdsourcing che memorizza gli ID di entità e categorie e li collega agli articoli multilingue di Wikipedia [10]. In YAGO3 entità e categorie hanno un prefisso che indica la lingua della relativa edizione di Wikipedia.

```
"de/Amerikanische Sänger"  hasTranslation "American
                        Singer"

     de/Elvis     hasTranslation          Elvis
```

Qualora si estraggano entità senza un corrispondente inglese, la lista delle lingue viene consultata. Tale lista stabilisce delle precedenze, ad esempio: `<English, French, Italian, German>`. Se si intende inserire l'entità "comune di Bologna" estratta dall'articolo italiano di Wikipedia, prima di inserirla come entità dell'articolo italiano bisogna controllare che essa non compaia nelle edizioni inglese e francese. Ciò permette di connettere ogni articolo sulla stessa entità a un *unique entity name*. L'insieme di tutti gli unique entity name costituisce l'insieme di entità di YAGO3.

4.1.2 Estrazione dei fatti

Si considerino i seguenti fatti: il primo contenuto in YAGO, il secondo estratto da un articolo di Wikipedia in lingua tedesca:

```
      1: Elvis  marriedTo Priscilla_Presley
2: de/Elvis    de/heirat de/Priscilla_Presley.
```

Entità e relazioni sono differenti e per la KB i fatti risultano indipendenti; tuttavia il lettore può intuire facilmente che i due fatti contengono la stessa informazione. È possibile innanzitutto tradurre le entità con il prefisso "`de/`" trovando il corrispondente unique name.

```
Elvis     de/heirat Priscilla_Presley
```

È necessario quindi stabilire se esista una effettiva corrispondenza tra `de/heirat`, estratto dalla infobox dell'articolo, e la relazione `marriedTo` esistente in YAGO.
Viene fornita una descrizione formale su come avvenga l'estrazione di fatti dalle varie versioni di Wikipedia.

Sia F_a l'insieme delle coppie soggetto-oggetto che compaiono in una edizione linguistica di Wikipedia collegate da un certo attributo *a* della relativa infobox (la coppia `Elvis-Priscilla_Presley` è uno degli elementi dell'insieme $F_{de/heirat}$). Sia E_r l'insieme delle coppie soggetto-oggetto che compaiono in YAGO collegate tramite una data relazione *r* (la coppia `Elvis-Priscilla_Presley` è uno degli elementi dell'insieme $E_{marriedTo}$). È possibile sapere se *a* corrisponde a *r* studiando gli insiemi F_a e E_r e osservando quante coppie essi abbiano in comune. Si consideri *match* tra *a* e *r* qualunque soggetto x che abbia un oggetto in comune in F_a e in E_r. L'insieme dei match tra *a* e *r* è definito:

$$matches(F_a,E_r) = \pi_{subj}(F_a \cap E_r).$$

È stato sviluppato un extractor per questa mappatura, chiamato *Attribute Matcher*. Applicato all'esempio citato, esso restituirebbe:

```
de/heirat hasTranslation marriedTo.
```

Tali corrispondenze vengono usate da un *Attribute Mapper* che genera un nuovo fatto per YAGO:

```
Elvis marriedTo Priscilla_Presley.
```

Il fatto non viene aggiunto alla KB se, come in questo esempio, ne faceva già parte. Tuttavia è stato possibile arricchire YAGO con un milione di nuovi fatti provenienti da edizioni non inglesi di Wikipedia.

4.1.3 Costruzione della tassonomia

Nel capitolo 2 è stata esaminata la struttura tassonomica di YAGO. Sappiamo ad esempio che `Elvis` è nella category `American Singers`. Infatti nell' edizione inglese di Wikipedia le categorie cui appartengono gli articoli sono identificate tramite la parola "category".

```
Elvis inCategory "American Singers"
```

Le altre versioni di Wikipedia si riferiscono alle categorie con altri termini: l'edizione tedesca associa `Elvis` alla `Kategorie Amerikanische Sänger`. Tale informazione è fornita da Wikidata e consente al *Category Extractor* di estrarre il fatto:

```
de/Elvis inCategory "de/Amerikanische Sänger".
```

Infine è possibile tradurre entità e categorie ai rispettivi unique name, ottenendo il fatto:

```
Elvis inCategory "American Singers".
```

4.2 Valutazione generale

4.2.1 Dimensione

La figura 15 illustra il numero totale delle entità e dei fatti che popolano la base di conoscenza. Ogni entità è considerata solo per la prima lingua della lista delle preferenze, malgrado informazioni su di essa possano essere state estratte da altre edizioni linguistiche. È stato possibile arricchire la KB con 1 milione di nuove entità tramite l'estrazione multilinguistica di YAGO3 [10].

Language	Entities	Facts
en	3,420,126	6,587,175
de	349,352	984,83
fr	255,063	549,321
nl	204,566	249,905
it	67,33	148,268
es	118,467	43,271
ro	11,024	12,871
pl	103,44	235,357
ar	50,295	98,285
fa	16,243	27,041
total	4,595,906	8,936,324

Figura 15: Entità e fatti di YAGO3 [10].

4.2.2 Precisione

La figura 16 mostra la precisione risultante per ogni lingua. È stato considerato un intervallo di confidenza del 16%.

Lingua	Precisione	Recall	F
ar	100	73	85
de	100	37	54
es	96	19	32
fa	100	49	66
fr	100	16	27
it	100	7	12
nl	100	19	32
pl	95	10	19
ro	96	52	67

Figura 16: Valutazione della precisione di YAGO3 [10].

4.3 Applicazione con Le Monde

Lo sviluppo di KB come YAGO ha permesso di archiviare grandi moli di dati testuali disponibili sul Web in forma strutturata. I dati strutturati raccolti nei computer rappresentano una nuova opportunità: è possibile utilizzarli per comprendere le informazioni in forma non-strutturata [8]. Alcuni studiosi hanno proposto di utilizzare le informazioni relative ai fatti per interpretare i documenti testuali prodotti da vari utenti del Web. Esistono progetti che hanno fatto progressi in questo ambito focalizzandosi sull'analisi della frequenza dei termini per studiare l'evoluzione di una lingua nel tempo.

Gli esperti del Max Planck Institute for Computer Science hanno voluto rappresentare un'avanguardia andando oltre lo studio della frequenza delle parole nei brani. A tale scopo hanno usufruito di una sezione dell'archivio di Le Monde, noto quotidiano francese. La raccolta parte dalle edizioni del 1944 e copre il periodo postbellico, la fine del colonialismo, quindi le guerre, la politica, gli sport, la cultura e l'economia fino all'anno 1986 [8]. Essa conta un totale di 502781 articoli [8] caratterizzati da: titolo, data di pubblicazione, testo. Un'analisi approfondita dei trend di questi anni è stata possibile creando dei collegamenti tra i nomi ricorrenti negli articoli e le entità della KB. È stato possibile effettuare analisi statistiche e tracciare gli sviluppi avvenuti nel tempo, dimostrando fino a che punto i dati strutturati possano essere utilizzati.

È stato in primo luogo necessario identificare le entità negli articoli del quotidiano. Sono stati utilizzati metodi statistici per trovare gli entity name e classificarli in persone, luoghi, organizzazioni. Le entità estratte dal brano sono state poi disambiguate attraverso un metodo innovativo e poco costoso: sono state raccolte statistiche

sulla frequenza di particolari espressioni degli articoli del quotidiano che si riferissero a un corrispondente articolo di Wikipedia francese. In questa fase sono state trascurate frasi poco frequenti o insolite. Gli articoli di Wikipedia sono stati quindi associati alle entità di YAGO grazie alle informazioni plurilinguistiche possedute dalla KB. I risultati di questa operazione sono raccolti in una tabella che indica, per ogni articolo, quali entità di YAGO vi compaiano. Inoltre molti articoli di Le Monde si riferiscono a un avvenimento con una certa dimensione spaziale. È stato possibile identificare in quali stati gli eventi fossero avvenuti. La tecnica utilizzata a tale scopo ipotizza che il luogo di accadimento sia quello più ricorrente nel corpo dell'articolo; si considera inoltre che citare una certa località sia equivalente a citare lo stato cui essa appartiene. Questa analisi ha permesso di creare una tabella che contiene, per ogni articolo, l'ubicazione dell'evento descritto. Le due tabelle ottenute, insieme a quella che mostra la data di pubblicazione di ogni articolo, sono state sufficienti per raggiungere gli obiettivi del progetto. Gli studiosi hanno compiuto analisi numeriche rispetto a varie dimensioni: alcune di esse vengono riportate di seguito per mostrare il valore aggiunto che i dati strutturati offrono nell'analisi dei dati non strutturati.

Grazie all'aiuto di YAGO gli studiosi hanno conosciuto l'occorrenza di uomini, donne, politici e politiche menzionati nel quotidiano. I risultati di questa analisi sono mostrati nella figura 17. Malgrado sia visibile una leggera crescita nella presenza di nomi di donne e politiche nel corso degli anni, il divario con la ricorrenza di soggetti maschili continua a essere significativa.

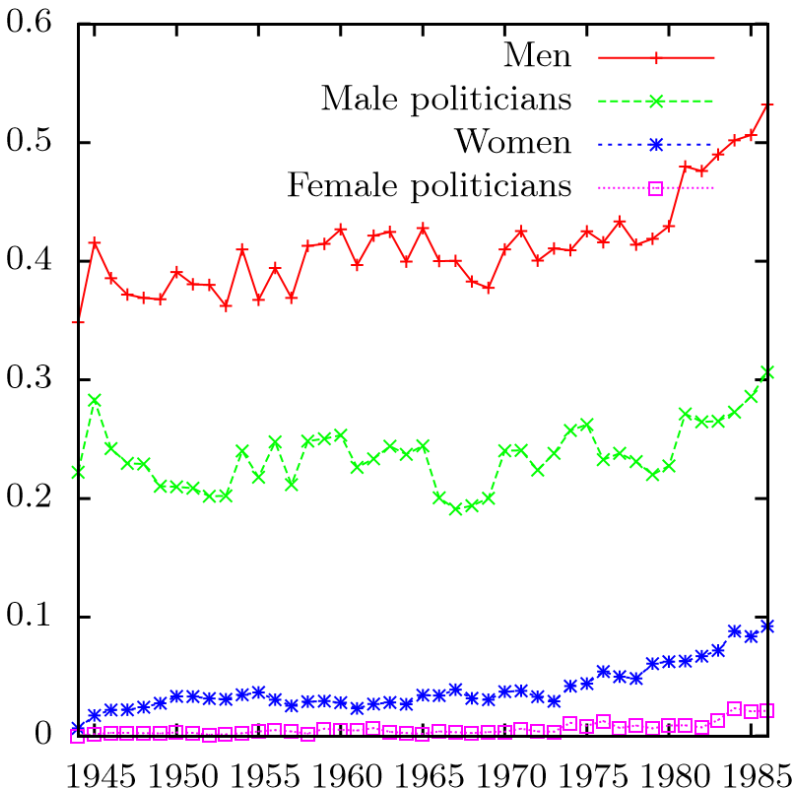

Figura 17: Riferimenti a uomini e donne [8].

Il lavoro svolto ha permesso inoltre di rappresentare l'andamento dell'età media di cantanti, politici e musicisti. Nella figura 18 è mostrato il risultato di tale analisi: si nota come l'età media dei cantanti sia significativamente inferiore a quella dei politici, malgrado un incremento sia avvenuto nel corso dei decenni.

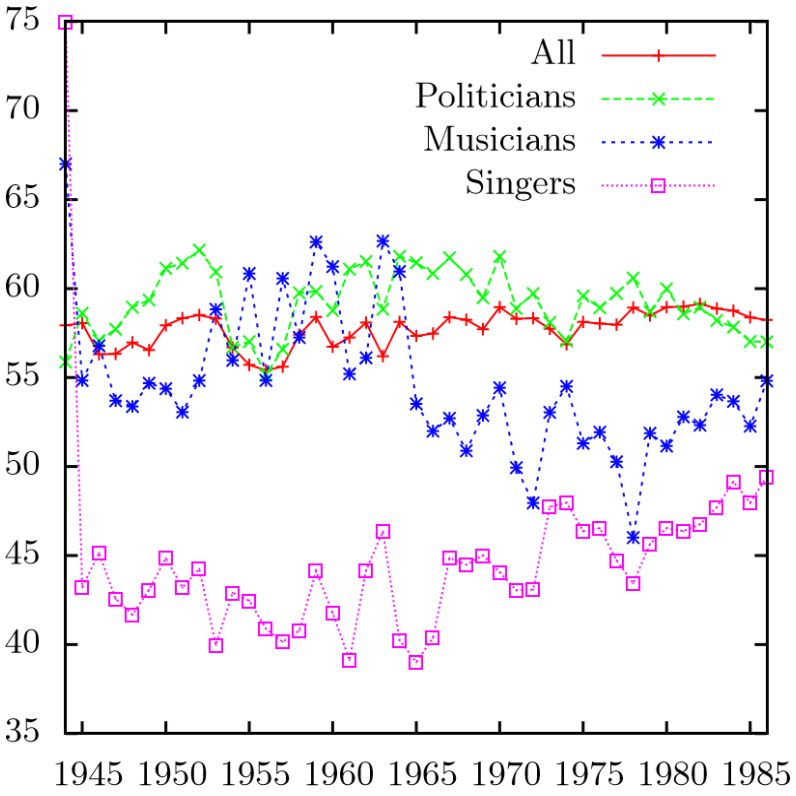

Figura 18: Età media dei soggetti citati [8].

La figura 19 mostra infine con quale ricorrenza i vari stati, raggruppati per continente, siano citati negli articoli di Le Monde. Nel periodo postbellico circa il 40% degli articoli citavano paesi europei, mentre gli stati africani sono stati nominati frequentemente negli anni '60, periodo in cui molti di essi hanno ottenuto l'indipendenza.

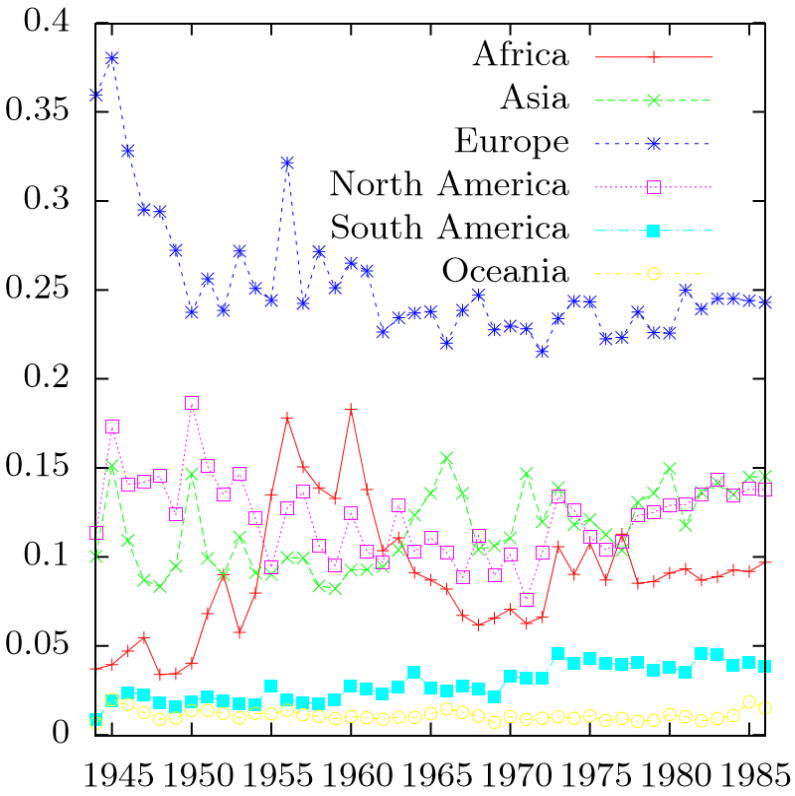

Figura 19: Paesi menzionati raggruppati per continente [8].

Al fine di illustrare in che modo la conoscenza spaziale può essere usata per esplorare documenti testuali, è stato fatto uno studio su quante delle persone ricorrenti nel giornale fossero nate nella capitale del paese di appartenenza. La figura 20 mostra il risultato di tale analisi. I paesi con colore rosso sono quelli in cui i personaggi rinomati sono nati nelle capitali, quelli colorati di blu hanno la caratteristica opposta. Il 98% delle persone norvegesi citate dal giornale sono nate a Oslo, mentre solo il 7% degli americani analizzati sono nati a Washington.

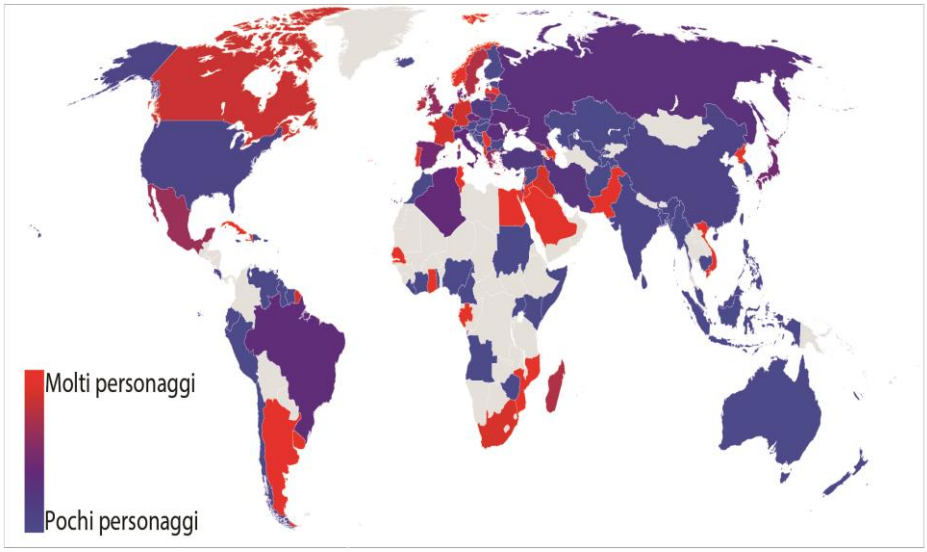

Figura 20: Ricorrenza di personaggi famosi nati nelle capitali [8].

Capitolo 5 - Ambiti di ricerca sulle KB

Questo capitolo esamina alcuni aspetti della gestione delle basi di conoscenza. In particolare vengono trattati la protezione dal plagio e l'allineamento tra diverse KB. Viene utilizzata la base di conoscenza YAGO a fini esemplificativi.

5.1 Paris

Esistono alcune KB specifiche per determinati ambiti: musica, cinema, dati geografici, pubblicazioni scientifiche sono alcuni esempi. Altre KB sono generiche e, come YAGO, coprono vari campi della conoscenza. Molte volte queste basi di conoscenza sono complementari: una KB generica può conoscere un certo enzima e chi lo ha scoperto mentre un database biologico ne conosce le proprietà. Tuttavia le basi di conoscenza utilizzano spesso identifier differenti per la medesima entità e risulta difficile collegare e raccogliere informazioni di diverse KB. Tornando all'esempio, è difficile scoprire chi ha scoperto l'enzima con determinate proprietà. Lo stesso problema esiste per le relazioni: YAGO afferma che `Elvis wasBornIn Tupelo`, un'altra KB potrebbe affermare che `Tupelo` è il `placeOfBirth` di `Elvis`. L'approccio probabilistico *PARIS* permette di risolvere entrambi i problemi di allineamento operando un match simultaneo tra entità, classi o relazioni equivalenti tra più KB [14]. La sigla PARIS sta per "Probabilistic Alignment of Relations, Instances and Schema" [14].
Questo approccio si basa su un'intuizione: le uguaglianze tra istanze e relazioni si determinano vicendevolmente. Tale intuizione diventa una connessione effettiva quando la relazione è una *relazione*

funzionale. Una relazione è funzionale se accetta, per ogni primo argomento, al più un secondo argomento [14].
La relazione wasBornIn è una relazione funzionale; la relazione hasSon non è una relazione funzionale poiché una persona può avere più di un figlio.

Conseguentemente è detta relazione funzionale *inversa* ogni relazione che abbia al più un primo argomento per ogni secondo argomento (si pensi alla relazione hasSon). Pertanto due istanze *x* e *x'* sono equivalenti se condividono lo stesso secondo argomento di una relazione funzionale inversa [2]:

$$\exists r, y, y' : r(x, y) \wedge r(x', y') \wedge y \equiv y' \wedge r \textit{ inv. Functional} \quad \Rightarrow x \equiv x'.$$

Le basi di conoscenza contengono spesso errori e per questo è stato progettato un modello probabilistico per rilassare le regole logiche: si tratta di sollevare temporaneamente le diverse KB dalle relative regole logiche al fine di ricavare la corrispondenza tra entità, relazioni e classi e effettuare quindi l'allineamento tra KB. Se non venissero trascurate le regole sarebbero possibili solo alcune delle corrispondenze esistenti tra entità a cause di vincoli formali non rispettati.

5.2 Watermarking

La maggior parte delle basi di conoscenza sono disponibili gratuitamente. Tuttavia il loro utilizzo è spesso regolato da licenze: se un utente ne ripubblica alcuni dati deve attribuirli ai creatori della KB per non incorrere nel reato di plagio [2].

Un soggetto che ripubblica un'ontologia o parte di essa in maniera difforme alle licenze è detto "attacker". Egli potrebbe pubblicare la suddetta ontologia sotto il proprio nome o usarne una sezione senza riconoscerla ai creatori. La ontologia che funge da sorgente è detta "original ontology" e quella ripubblicata è detta "suspect ontology"[16].

Le asserzioni ontologiche sono solitamente conoscenza comune sul mondo che ci circonda, perciò un soggetto può ottenere la stessa informazione da più fonti: potrebbe egli stesso rivendicare i diritti su tali dati a danno dei creatori della KB. Perciò si solleva la domanda su come sia possibile provare che un soggetto abbia ripubblicato illegalmente i dati di una KB. Tale problema può essere affrontato tramite il *digital Watermarking*, o "filigranatura digitale". Si tratta di un approccio sofisticato che introduce piccole modifiche alla original ontology per determinare se una suspect ontology contenga dati derivanti dalla original ontology [16].

A tale scopo è necessario:

- Formalizzare il problema della ripubblicazione di dati ontologici
- Realizzare un algoritmo di Watermarking per le ontologie
- Effettuare esperimenti come prova della validità di tale approccio.

Le tecniche di Watermarking mirano a nascondere alcune informazioni rilevanti in un data set. Trovare le stesse informazioni in un data set sospetto rappresenta una prova della proprietà. Molti progetti per i database relazionali utilizzano il Watermarking eseguendo delle alterazioni volontarie sui dati: tale approccio potrebbe essere esteso alle KB. Inevitabilmente l'alterazione rende inferiore la precisione dell'ontologia e per questo motivo è stato

sviluppato un metodo alternativo, specifico per il semantic Web. È stato proposto di contrassegnare un'ontologia rimuovendo attentamente alcune affermazioni prima della pubblicazione. L'assenza sospetta di queste in un'altra ontologia, se coincidente varie volte, vale come prova di plagio.

Sono stati quindi sviluppati due approcci per YAGO: Additive Watermarking e Subtractive Watermarking [2].

- Additive Watermarking. Alcuni false affermazioni dette "fake fact" vengono aggiunte alla KB. Se questi appaiono in un'altra KB, allora questa ha probabilmente usato informazioni della base di conoscenza originale. I fake fact devono essere abbastanza verosimili da non essere riconosciuti da una macchina o da un operatore umano. La maggiore obiezione a questo approccio è che esso peggiora la qualità della KB. È vero però che il Watermarking è sempre un compromesso tra qualità dei dati e abilità nel riconoscere la provenienza di questi. Tuttavia le tecniche utilizzate aggiungono un numero di fake fact molto ristretto. Le KB di grandi dimensioni e costruite automaticamente contengono inevitabilmente alcuni fatti errati. Si prenda come esempio YAGO che conosce milioni di fatti e la cui correttezza è del 95%: risultano alcune migliaia di fatti errati e perciò è sufficiente solo una piccola aggiunta di fake fact.

- Subtractive Watermarking. Alcune affermazioni vengono rimosse dalla KB prima della pubblicazione. Si crea un pattern di lacune: si possiede una prova di plagio qualora lo stesso pattern appaia in un'altra KB. Questa procedura non penalizza la precisione dell'original ontology ed è in accordo

con l'assunzione "Most ontologies are incomplete", valida per il Semantic Web. La completezza risulta tuttavia penalizzata [16].

Infine la Figura 21 mostra, per i due approcci descritti, la precisione e la dimensione. Utilizzando queste tecniche infatti la base di conoscenza risulta modificata e perciò anche i parametri sono alterati. Nel caso di Additive Watermarking si aggiungono informazioni alla KB e la sua dimensione aumenta. Tali informazioni sono tuttavia false e per questo la precisione della KB diminuisce. Si pensi a una KB contenente 10 documenti di cui solo 7 rilevanti. La sua precisione è

$$\textit{Precisione inziale caso Additive Watermarking} = P_{ia}=7/10.$$

Si voglia utilizzare su questa la tecnica di Additive Watermarking, aggiungendo 1 fake fact alla raccolta. I documenti contenuti diventano perciò 11 ma resta costante il numero di documenti rilevanti, poiché il fake fact è falso per definizione

$$\textit{Precisione finale caso Additive Watermarking} = P_{fa}=7/11.$$

Risulta $P_{fa}<P_{ia}$: la precisione diminuisce con l'approccio Additive Watermarking.
Nel caso di Subtractive Watermarking vengono eliminati elementi della KB e la sua dimensione risulta minore. La precisione diminuisce allo stesso modo poiché entrambi i fattori determinanti calano in maniera identica.
Si pensi alla KB precedente, caratterizzata da

$$\textit{Precisione inziale caso Subtractive Watermarking} = P_{is}=7/10$$

e si voglia utilizzare su questa la tecnica di Subtractive Watermarking. Eliminando 1 documento dalla raccolta la dimensione passa da valore 10 a valore 9. Il fatto sottratto è rilevante per la definizione stessa della tecnica perciò la precisione finale risulta:

Precisione finale caso Subtractive Watermarking = P_{fs}=6/9.

Si ha come risultato $P_{fs}<P_{is}$: la precisione diminuisce con l'approccio Subtractive Watermarking.

È possibile quindi affermare che entrambe le tecniche di Watermarking tutelano la KB originale a discapito della precisione: in un caso ciò avviene aumentandone la dimensione, nell'altro diminuendola.

	Precisione	Dimensione
Additive Watermarking	minore	maggiore
Subtractive Watermarking	minore	minore

Figura 21: Additive Watermarking e Subtractive Watermarking.

Conclusioni

Il presente elaborato è stato realizzato principalmente grazie al materiale fornito dagli sviluppatori del progetto YAGO, disponibile online gratuitamente in lingua inglese, e rappresenta una delle poche trattazioni sul tema in lingua italiana reperibili sul Web.
La scelta dell'oggetto di tesi è dovuta a due motivi principali: l'interesse personale e l'innovazione rappresentata dal tema.
Ulteriori progressi in questo campo rappresenterebbero un'importantissima risorsa e fonte di conoscenza facilmente reperibile. È possibile pensare ai più diversi dati di tipo scientifico o letterario, riguardanti l'attualità, il governo o eventi accaduti, gli aggiornamenti in campo giuridico e politico. Moltissime sono le leggi emanate ogni giorno, le scoperte realizzate in ambito scientifico o i libri pubblicati nei vari paesi: se ognuno di questi dati rimanesse una singola scatola isolata dalle altre non vi si potrebbe accedere, privando l'utenza del Web di conoscenze non sostituibili e generando lacune di informazioni non colmabili.
Esistono infatti soggetti le cui abilità non permettono l'utilizzo ottimale delle risorse del Web. Qualora un soggetto fosse capace di utilizzare facilmente i motori di ricerca, esistono situazioni in cui raccogliere informazioni da fonti diverse diventa un'operazione eccessivamente onerosa per fornire risultati soddisfacenti.
Un'immensa mole di dati viene prodotta quotidianamente da un mondo in continuo e rapido cambiamento, così come è in continua crescita il numero di ricerche effettuate dagli utenti nel Web. Lo sviluppo di ontologie digitali a supporto della strutturazione e dell'interrogazione di dati permetterebbe una fruizione efficace e immediata della conoscenza ancora nascosta nei documenti presenti in rete.

Bibliografia

[1] Adolphs, P., Theobald, M., Schafer, U., Uszkoreit, H., & Weikum, G. (2011). Yago-qa: Answering questions by structured knowledge queries. In *Semantic Computing (ICSC), 2011 Fifth IEEE International Conference on*(pp. 158-161). IEEE.

[2] Amarilli, A., Galárraga, L., Preda, N., & Suchanek, F. M. (2014). Recent topics of research around the YAGO knowledge base. In *Asia-Pacific Web Conference* (pp. 1-12). Springer International Publishing.

[3] Fabian, M. S., K. Gjergji, and W. Gerhard. (2007). "Yago: A core of semantic knowledge unifying wordnet and wikipedia." *16th International World Wide Web Conference, WWW*.

[4] Fabian, M. S., and W. Gerhard. (2008). “Searching for knowledge instead of web sites.”.

[5] Galárraga, Luis Antonio, et al. (2013) "AMIE: association rule mining under incomplete evidence in ontological knowledge bases." *Proceedings of the 22nd international conference on World Wide Web*. ACM.

[6] Hoffart, J., Suchanek, F. M., Berberich, K., Lewis-Kelham, E., De Melo, G., & Weikum, G. (2011). YAGO2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World wide web* (pp. 229-232). ACM.

[7] Hoffart, J., Suchanek, F. M., Berberich, K., & Weikum, G. (2013). YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, *194*, 28-61.
[8] Huet, Thomas, Joanna Biega, and Fabian M. Suchanek. (2013). "Mining history with le monde." *Proceedings of the 2013 workshop on Automated knowledge base construction*. ACM.
[9] Kasneci, G., Ramanath, M., Suchanek, F., & Weikum, G. (2009). The YAGO-NAGA approach to knowledge discovery. *ACM SIGMOD Record*, *37*(4), 41-47.
[10] Mahdisoltani, F., Biega, J., & Suchanek, F. (2014). Yago3: A knowledge base from multilingual wikipedias. In *7th Biennial Conference on Innovative Data Systems Research*. CIDR Conference.
[11] Martinez-Gil, J. (2015). Automated knowledge base management: A survey.*Computer Science Review*, *18*, 1-9.
[12] Radinsky, Kira, and Eric Horvitz. (2013). "Mining the web to predict future events."*Proceedings of the sixth ACM international conference on Web search and data mining*. ACM.
[13] Signore, Oreste. (2003). "Strutturare la conoscenza: XML, RDF, Semantic Web."*Proceedings of Clinical Knowledge 2003*.
[14] Suchanek, Fabian M., Serge Abiteboul, and Pierre Senellart. (2011). "Paris: Probabilistic alignment of relations, instances, and schema." *Proceedings of the VLDB Endowment* 5.3: 157-168.
[15] Suchanek, Fabian M., and David Gross-Amblard. (2012). "Adding fake facts to ontologies." *Proceedings of the 21st International Conference on World Wide Web*. ACM.
[16] Suchanek, Fabian M., David Gross-Amblard, and Serge Abiteboul. (2011). "Watermarking for

ontologies." *International Semantic Web Conference*. Springer Berlin Heidelberg.

[17] Suchanek, F. M., Kasneci, G., & Weikum, G. (2008). Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web*, *6*(3), 203-217.

[18] Suchanek, F., & Weikum, G. (2013). Knowledge harvesting in the big-data era. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data* (pp. 933-938). ACM.

[19] Wang, Y., Zhu, M., Qu, L., Spaniol, M., & Weikum, G. (2010). Timely yago: harvesting, querying, and visualizing temporal knowledge from wikipedia. In *Proceedings of the 13th International Conference on Extending Database Technology* (pp. 697-700). ACM.

[20] Weikum, Gerhard, Srikanta Bedathur, and Ralf Schenkel. (2010). "Temporal knowledge for timely intelligence." *International Workshop on Business Intelligence for the Real-Time Enterprise*. Springer Berlin Heidelberg.

[21] Weikum, G., Kasneci, G., Ramanath, M., & Suchanek, F. (2009). Database and information-retrieval methods for knowledge discovery. *Communications of the ACM*, *52*(4), 56-64.

[22] Weikum, G., & Theobald, M. (2010). From information to knowledge: harvesting entities and relationships from web sources. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* (pp. 65-76). ACM.