

UNIVERSITA' DELLA CALABRIA

Dipartimento di ingegneria Informatica, Modellistica, Elettronica e Sistemistica

Dottorato di Ricerca in

Ingegneria dei Sistemi e Informatica

Con il contributo di (Ente finanziatore)

FSE

CICLO

XXVI

Interrogare in linguaggio naturale una base di conoscenza Datalog

Un approccio basato sull'annotazione semantica dei predicati

Settore Scientifico Disciplinare ITC

Coordinatore: Ch.mo Prof. Sergio Greco

Firma Sergio Greco

Supervisore/Tutor: Ch.mo Prof. Francesco Scarcello

Firma Francesco Scarcello

Dottorando: Dott. Fabrizio Lo Scudo

Firma Fabrizio Lo Scudo

Indice

Introduzione

I recenti sviluppi nelle tecnologie per il riconoscimento del linguaggio naturale e della voce umana permettono una sempre maggiore interazione fra gli utenti e i dispositivi elettronici. A differenza di qualche anno fa, infatti, è oggi possibile utilizzare il linguaggio naturale per ottenere informazioni su specifici domini d'interesse con una precisione molto alta.

Questa tesi ha un duplice obiettivo: in primo luogo presentare un quadro abbastanza ampio, ancorché non esaustivo, dei numerosissimi studi, condotti in vari campi, alla base delle moderne tecnologie di interazione col linguaggio naturale; in secondo luogo descrivere una metodologia (ed una sua possibile implementazione) per realizzare un'interfaccia naturale per basi di conoscenza codificate nel linguaggio logico Datalog. Oltre all'evidente salto di qualità nelle possibili interazioni con tali basi di conoscenza, la disponibilità di una interfaccia in linguaggio naturale potrebbe alleviare i grandi (e tuttora largamente irrisolti) problemi di integrazione dei sistemi informativi in organizzazioni complesse (ma in realtà anche e forse ancor più in organizzazioni semplici e per questo meno strutturate). In ogni ambito vengono infatti usate applicazioni specifiche che raramente dialogano con le altre applicazioni della stessa organizzazione. Anche utenti esperti difficilmente riescono a fruire di tali informazioni in modo integrato, a meno di non conoscere precisamente tutti gli schemi dei dati sottostanti le base di conoscenza (o di dati) a loro disposizione. Idealmente, la capacità di accedere a tali informazioni mediante l'uso del linguaggio naturale potrebbe costituire una sorta di interfaccia universale per l'accesso a tutte le informazioni a disposizione.

È evidente che si tratta di un problema complesso e che richiede competenze trasversali, che vanno dall'analisi del linguaggio naturale alla rappresentazione della conoscenza ed alle tecniche per il ragionamento. La presente tesi non fornisce ancora una risposta completa per il pieno raggiungimento di questo obiettivo (di medio-lungo termine), ma descrive gli strumenti principali e pone le basi metodologiche per progredire verso la meta.

Nel corso degli anni, lo studio dei linguaggi naturali ha evidenziato come la complessità dell'analisi si estenda lungo un intricato interpiano formato

dalle tante componenti che entrano in gioco durante l'azione della comunicazione linguistica. Molto lavoro è stato svolto sugli aspetti fondamentali del linguaggio, suddivisi successivamente in fonologia, morfologia, sintassi, semantica e pragmatismo solo per citarne alcuni. Ciononostante il linguaggio naturale rimane intrinsecamente complesso nella sua totalità.

Si inizi col considerare le parole come le unità atomiche del significato. Sfortunatamente il processo che lega una parola, sia essa pronunciata o scritta, ad un riferimento o ad un significato non segue regole sistematiche o semanticamente naturali. Il concetto di onomatopeico è ormai considerato alla stregua di un semplice mito, nonostante si possano trovare qua e là esempi di similarità di significato fra parole fonologicamente simili. Infine, la discretizzazione di parole e morfemi[1] in fonemi non attribuisce a quest'ultimi un significato semantico o grammaticale.

La complessità cresce maggiormente con la possibilità per le parole di combinarsi fra di loro attraverso un insieme di regole sintattiche per formare enunciati. Il senso inteso è (parzialmente) determinato sia dalla struttura delle parole (p.es. particolari suffissi) sia dalla loro posizione all'interno della frase. Lo scopo della sintassi è quindi quello di ridurre l'ambiguità in un linguaggio che altrimenti sarebbe semplicemente una collezione di termini, il cui utilizzo, non essendo regolato, sarebbe a completa discrezione di colui che sta parlando. È importante notare, tuttavia, come non tutte le frasi sintatticamente corrette possiedano un significato di senso compiuto. È compito della semantica guidare una "logica" formazione del senso di una frase. La semantica descrive infatti il modo in cui un'interpretazione, intesa come il risultato di una combinazione del significato delle singole parole, è costruita sulla base della struttura sintattica della frase. L'analisi fonologica, morfologica e sintattica sono gli strumenti di base sui quali quale fondare l'interpretazione di un espressione. Essi facilitano l'individuazione dei ruoli di ogni costituente e servono ad esplicitare i legami che sussistono fra di loro. Per esempio, nella frase "*James gives Alise a book.*" la strutturazione sintattica permette agevolmente di individuare *chi* compie l'azione, *chi* la riceve e *che cosa* sia l'oggetto dell'azione. Ciononostante, altri aspetti come punteggiatura, intonazione, o risoluzione delle riprese anaforiche, richiedono ulteriore conoscenza per una corretta interpretazione. Pronomi, proposizioni ellittiche, e altre forme di ambiguità o di *sotto-specificazione* necessitano tutte di una certa quantità di informazione extra-grammaticale per costruire l'interpretazione più opportuna per il contesto dato. Risolvere certe ambiguità va ben oltre la conoscenza grammaticale richiesta per licenziare certe espressioni come conformi ad un determinato linguaggio: riuscire a ricostruire il messaggio contenuto in una frase richiede un processo di inferenza molto più generale.

[1] Un morfema è un unità atomica che può essere combinata con altri morfemi per definire la struttura del significato di una parola. Ad esempio il verbo *to send* ha come morfema *send*, che a sua volta può essere combinato con altri morfemi per produrre la sua coniugazione: *sends, sending, etc.*

In questo lavoro siamo però interessati ad interagire con basi di conoscenza e, quindi, con informazioni che per ipotesi sono ben formalizzate. La presenza di un linguaggio formale che modella gli aspetti rilevanti di un fenomeno o di un dominio di interesse (concetti, relazioni, proprietà, fatti, regole etc.) è di primaria importanza per i processi di inferenza automatica. Intendiamo anche sfruttare il fatto che, tipicamente, la modellazione di un dominio di interesse inizia con una descrizione informale del dominio, normalmente espressa in linguaggio naturale, alla quale possono essere affiancate descrizioni formali. Si procede poi prendendo in considerazione le finalità di utilizzo del sistema (o dei sistemi): come sarà usata la conoscenza, per quali finalità, quali figure professionali si interfacceranno con essa, ecc. È quindi presente una visione comune della struttura dell'informazione tra un gruppo di uomini e/o agenti software, che può permettere l'uso di tale conoscenza del dominio anche per le analisi di cui ci occupiamo in questo lavoro di tesi. Se non già presente, assumiamo che una rappresentazione addizionale delle informazioni in linguaggio naturale possa facilmente essere aggiunta.

Il formalismo che si è scelto di usare per la rappresentazione della conoscenza, ma anche per la codifica dell'interrogazione da effettuare a tale base di conoscenza, è la programmazione logica, più precisamente il framework noto come *Answer Set Programming* (ASP), per il quale faremo riferimento, ove necessario per aspetti implementativi, al software DLV (vedi Capitolo 6). Questo formalismo è caratterizzato da una notevole capacità espressiva ed è adeguato per i nostri fini di interrogazione che non sono web-oriented, bensì concentrati su basi di conoscenza per le quali si può assumere valida la cosiddetta *assunzione di mondo chiuso* e l'unicità dei nomi (non vi possono essere due individui con lo stesso identificativo).

Assumendo quindi una base di conoscenza KB codificata in ASP, questo lavoro propone un approccio rivolto all'arricchimento di KB con una serie di meta-informazioni così da renderla interrogabile in linguaggio naturale. Più nel dettaglio il tipo di meta-informazione da affiancare può essere assimilato ad una particolare tipologia di annotazione che prevede la definizione per ogni predicato di una struttura di supporto contenente:

- un'opportuna descrizione in linguaggio naturale che descrive il ruolo dei vari attributi, visti come entità, coinvolti in quel predicato;
- l'utilizzo di vettori di parole per l'individuazione delle entità del predicato all'interno di uno spazio vettoriale;
- il riferimento delle entità all'interno delle reti semantiche: BabelNet e ConceptNet;
- la codifica di informazioni di natura ontologico-funzionale.

Queste meta-informazioni svolgono un ruolo importante durante la fase di allineamento fra i concetti dell'interrogazione e quelli della basi di conoscenza. L'utilizzo di risorse lessicali e semantiche permette di individuare collegamenti e relazioni tra i predicati oltre quelli definiti in KB. Sopratutto l'impiego di misure di similarità semantica costruite su modelli neurali per il linguaggio

permette al sistema di estendere la sua copertura oltre i limiti imposti dalla sua terminologia o da una sua semplice estensione attraverso l'uso di sinonimi.

Terminata la fase di arricchimento è possibile interrogare la base di conoscenza con domande poste in linguaggio naturale. La generazione di una risposta ad una interrogazione comprende i seguenti sotto-problemi:

1. elaborazione sintattica della domanda: in questa fase si individuano le parti del discorso (POS) e le dipendenze che sussistono fra i termini della frase;
2. individuazione della tipologia di interrogazione: sulla base di regole euristiche si classifica la domanda e si definisce il tipo di risposta da restituire in output;
3. forma logica preliminare: si costruisce un programma Datalog utilizzando i termini presenti nell'interrogazione in modo sostanzialmente indipendente dalle reali risorse a disposizione;
4. allineamento dei concetti e dei predicati: utilizzando le meta-informazioni della base di conoscenza si tenta di individuare i concetti presenti nell'interrogazione fra quelli della KB;

L'analisi sintattica è eseguita utilizzando gli strumenti messi a disposizione dal centro di linguistica computazionale dell'università di Stanford. La classificazione della domanda si basa invece su un insieme di regole, costruite seguendo le metodologie presenti in letteratura. Mentre queste due fasi seguono approcci standard, le fasi successive di costruzione della forma logica e di allineamento dell'informazione rappresentano il principale contribuito di questo lavoro.

L'interpretazione della domanda inizia con la definizione di un programma logico a partire dall'analisi dell'annotazione sintattica della domanda stessa. Il grafo delle dipendenze è visitato partendo dal nodo radice e si costruiscono per ogni livello una o più regole logiche utilizzando nuovi simboli di predicato. Le regole così generate risultano per costruzione incomplete, poiché mancano del loro riferimento con la base di conoscenza KB. Attraverso l'utilizzo di una misura di similarità semantica, definita così da tener in considerazione anche i contesti, si tenta di selezionare l'entità più opportuna presente in KB. Si noti come l'utilizzo di una misura di similarità semantica, basata su vettori di parole, sia qui possibile grazie alla presenza delle meta-informazioni fornite in fase di arricchimento della base di conoscenza. Una volta arricchite tutte le regole, costruite a partire dalle relazioni di dipendenza, e risolte le eventuali entità rimaste fuori, il programma è pronto per essere eseguito sulle informazioni contenute nella base di conoscenza. Gli insiemi di risposta (answer set) generati, se presenti, saranno la risposta alla domanda posta.

Nel caso in cui le misure di similarità utilizzate nell'elaborazione preliminare dell'interrogazione non raggiungano una data soglia di accettabilità, il sistema può anche dichiararsi direttamente incapace di rispondere all'interrogazione posta. È infatti importante notare che si intende rispondere solo alle interrogazioni che sono perfettamente "coperte" dalla base di conoscenza,

pertanto vale l'assunzione di mondo chiuso che, in qualche modo, possiamo considerare non solo per i dati ma anche per i concetti coinvolti (se un concetto menzionato nell'interrogazione non è presente in KB allora non possiamo esprimerci in merito e lo notifichiamo all'utente).

Per quanto attiene una concreta adozione dell'approccio proposto in applicazioni reali, osserviamo che predicati e regole della base di conoscenza rappresentano implicitamente, in modo compatto, una serie limitata di possibili interrogazioni a KB. Tipicamente tali regole non sono molte, rispetto alla mole dei dati presenti nella parte estensionale (EDB), e possiamo ragionevolmente assumere che possano essere annotate, in modo semi-automatico (cioè parzialmente manuale e comunque supervisionato), per legare in modo più preciso i termini ed i loro ruoli nell'IDB della base di conoscenza con i concetti presenti nelle risorse semantiche a disposizione. In altri termini, sfruttiamo il fatto che i dati sono strutturati e, in termini di basi di dati, stiamo annotando solo lo schema (ed eventualmente i vincoli) della base di dati per ottenere automaticamente la caratterizzazione di tutto ciò che vi è memorizzato. Ancora, dal punto di vista della complessità computazionale della metodologia qui proposta, osserviamo che la fase di annotazione di KB avviene *una tantum, off-line*, mentre occorre essere veloci nella fase di interpretazione dell'interrogazione. Per questo motivo il carico computazionale è per lo più spostato nella prima fase, secondo un approccio "compilativo" che estenda la base di conoscenza con le altre risorse semantiche a disposizione.

Il lavoro di tesi è organizzato come segue.

Il primo capitolo introduce i più famosi formalismi utilizzati per verificare sul piano sintattico la correttezza delle espressioni linguistiche. Si inizia discutendo la teoria di Chomsky e le sue influenze sulle teorie successive, fra le quali sono discusse le grammatiche basate su categorie e su dipendenze. Di queste ultime sono poi riportate alcune recenti estensioni e viene discussa una tecnica per elaborare il testo basata sulle reti neurali.

Nel secondo capitolo sono discussi invece i principali approcci alla trattazione della semantica sviluppati nel corso degli ultimi due secoli. Si caratterizza il problema dell'interpretazione, dapprima dalla prospettiva linguistica e successivamente da quella filosofica. Viene discusso inoltre l'approccio definito da Montague e alcune sue estensioni per rappresentare strutture più complesse, come il discorso. Infine è descritta la teoria di rappresentazione astratta del significato.

La descrizione dell'analisi distribuzionale del significato è lo scopo del terzo capitolo. Sono definiti i principi di base di tale approccio e discussi i più recenti modelli per il linguaggio. Questi modelli, conosciuti anche con il nome di modelli neurali, sono analizzati nelle varie sezioni cercando dove possibile di evidenziarne pregi e difetti.

Il quarto capitolo descrive le risorse lessicali e le ontologie che possono essere impiegate nelle applicazioni che coinvolgono il linguaggio naturale, differenziate in base alla metodologia adottata per la loro costruzione. Si fornisce una analisi delle maggiori reti semantiche e ontologie oggi disponibili.

Il problema di generare una risposta per una domanda posta in linguaggio naturale è definito nel quinto capitolo. Dopo aver definito la questione si introducono le diverse tipologie di domanda e si discute una potenziale architettura, approfondendo alcune delle sue parti principali.

Nel sesto capitolo si presentano i concetti di base riferiti alla sintassi e alla semantica della programmazione logica disgiuntiva, usata per l'ASP. Sono inoltre illustrate le potenzialità dell'ASP per la rappresentazione di problemi complessi.

Nel settimo capitolo è presentata la metodologia proposta per rispondere ad interrogazioni in linguaggio naturale poste a basi di conoscenza in ASP. Si descrive inizialmente l'annotazione da affiancare ad una base di conoscenza. Successivamente è presentata la procedura di costruzione del programma logico a partire dall'analisi delle dipendenze e viene discussa la fase di allineamento con i predicati della base di conoscenza. Gli esempi riportati in questo capitolo fanno riferimento ad una base di conoscenza reale, poi semplificata per motivi di presentazione, generata a partire dalle informazioni relative alle attività di ricerca condotte presso l'Università della Calabria e raccolte in una scheda, predisposta dal Ministero dell'Istruzione, dell'Università e della Ricerca, denominata *SUA-RD*.

L'ottavo capitolo, infine, riporta le conclusioni ed i lavori correnti e futuri necessari per ottenere una interfaccia in linguaggio naturale efficiente ed efficace, possibilmente anche per interrogazioni in lingua italiana.

1

Formalismi per l'analisi sintattica del linguaggio naturale

Per poter costruire un'interpretazione per espressioni del linguaggio naturale si devono necessariamente conoscere i suoi elementi e quali regole governino la loro composizione. Circoscrivendo la discussione unicamente agli aspetti sintattici, questo capitolo introduce i formalismi utilizzati per rappresentare e verificare la correttezza delle dinamiche del linguaggio. La maggior parte delle regole presentate qui non appartengono all'insieme di regole tipicamente insegnate nei primi anni di scuola. Esse non forniscono alcun genere d'informazione su come inserire elementi di punteggiatura o dividere un'infinitiva, ma sono rivolte invece a verificare l'adeguatezza di un'espressione linguistica ad una grammatica di riferimento.

Nel seguito si introdurranno i concetti alla base dell'analisi sintattica del linguaggio. Si inizierà con la teoria di Chomsky sui linguaggi e alcune sue estensioni. Saranno poi discusse le grammatiche basate su categorie e sulle dipendenze. Infine verranno trattati alcune recenti estensioni della grammatica delle dipendenze e delle tecniche per elaborare il testo secondo questa teoria.

1.1 Grammatica Generativa

Fra le teorie più influenti nello sviluppo della linguistica moderna quella sviluppata da Chomsky attorno alla metà del secolo scorso è forse la più importante. Nel corso degli anni questa teoria è stata indicata con diversi nomi: Transformational Grammar, Transformational Generative Grammar, Standard Theory solo per citarne alcuni. Tuttavia oggigiorno è uso comune riferirsi ad essa con il termine di Grammatica Generativa. Sotto questa denominazione si fanno rientrare ormai anche diverse teorie più recenti come Lexical-Functional Grammar (LFG) e Head-Driven Phrase Structure Grammar (HPSG).

La tesi fondante nella grammatica generativa è l'idea secondo la quale un insieme di regole generi l'inseme (possibilmente infinito) di espressioni grammaticalmente corrette. Queste regole, assunte innate fra le nostre abilità cognitive, sono l'oggetto ultimo della modellazione operata dalla teoria. E'

importante notare come ciò sia in netto contrasto con l'approccio strutturalistico proposto da Saussure e proseguito poi da Bloomfield, Wells e Harris; e fu proprio in contrapposizione al programma sviluppato da Harris che Chomsky iniziò a sviluppare la sua teoria.

Le differenze fra strutturalismo e grammatica generativa risiedono nell'intenzione di Chomsky di enfatizzare aspetti linguistici al fine di rendere il sistema ricorsivo. Lo strutturalismo invece concentrò la sua analisi sui diversi livelli del linguaggio all'interno dei quali fornire una definizione dei diversi aspetti linguistici.

Nell'approccio generativo i simboli, presenti all'interno delle regole della grammatica, sono suddivisi in due categorie fondamentali: *terminali* e *non-terminali*. Mentre i primi possono essere facilmente ricondotti agli elementi di qualche linguaggio d'interesse, o categorie lessicali, i secondi si riferiscono a categorie sintattiche, come *Sentence*, *Noun Phrase*, o *Verb Phrase*.

Basate su questi simboli, le regole hanno il compito di definire la struttura interna di una frase. Ogni regola è caratterizzata da un solo simbolo non-terminale, detto *testa* della regola, a sinistra del simbolo speciale $\rightarrow$ e da una sequenza di simboli che compongono il *corpo*. Un piccolo esempio di grammatica per la lingua inglese può essere il seguente:

1. $S \rightarrow NP\ VP$
2. $NP \rightarrow Det\ NP$
3. $NP \rightarrow Word$
4. $VP \rightarrow Vt\ NP$
5. $Det \rightarrow the$
6. $Word \rightarrow John$
7. $Word \rightarrow book$
8. $Vt \rightarrow likes$

Qui l'insieme di simboli non-terminali è $\{S, NP, VP, Word, Vt, Det\}$, mentre per quelli terminali si ha $\{John, likes, the, book\}$. L'interpretazione della prima regola, per esempio, definisce la categoria sintattica per *Sentence* come composta dalla concatenazione di due categorie sintattiche nel seguente ordine: *Noun Phrase* e *Verb Phrase*.

Generalmente parlando, ci si riferisce alle regole appena descritte come *context-free rules* poiché esse non prendono in considerazione il contesto nel quale operano. Con queste regole è possibile analizzare frasi tipo "*John likes the book*". Formalmente è possibile definire una grammatica *context-free* come

Definition 1.1. *Una grammatica context-free è una tupla $G = \langle V, \Sigma, P, S \rangle$, dove V è l'insieme finito di simboli non-terminali, Σ è l'alfabeto dei simboli terminali, $P \subseteq V \times (V \cup \Sigma)^*$ è l'insieme di regole e $S \in V$ è un simbolo speciale riservato a rappresentare l'elemento iniziale.*

Nell'approccio generativo l'obiettivo della grammatica è di licenziare tutte le strutture corrispondenti a enunciati grammaticalmente corretti. Cioè generare tutte le sequenze di simboli che possono ritenersi corrette e complete senza considerare il contesto. Tuttavia prima di poter generare un linguaggio occorre introdurre le nozioni di *forma* e di *derivazione*.

Definition 1.2. *Data una grammatica* $G = \langle V, \Sigma, P, S \rangle$, *G genera un insieme di forme* $(V \cup \Sigma)^*$. *Una forma* f_1 *deriva immediatamente una forma* f_2, *indicata con* $f_1 \Rightarrow f_2$, *se e solo se esistono* $\sigma, \sigma' \in (V \cup \Sigma)^*$ *tale che* $f_1 = \sigma A \sigma'$, $f_2 = \sigma \hat{\sigma} \sigma'$ *e* $A \rightarrow \hat{\sigma}$ *è una regola in* P.

Dalla chiusura transitiva della relazione di forma appena descritta è possibile definire la relazione di derivazione come:

Definition 1.3. *La derivazione* $f_1 \Rightarrow *f_n$ *è definita ricorsivamente come:* $f_1 \Rightarrow *f_n$ *se* $f_1 = f_n$, *o se* $f_1 \Rightarrow f_x$ *e* $f_x \Rightarrow *f_n$.

Il linguaggio definito sulla base di queste relazioni è il linguaggio licenziato dalla grammatica.

Utilizzando la grammatica definita nell'esempio precedente, l'analisi della frase "*John likes the book*" produce la seguente espressione:

$$S(NP(Word\ John)\ VP((Vt\ likes)\ NP((Det\ the)(Word\ book))))$$

In generale non ci sono restrizioni sul corpo di una regola. Tuttavia è pratica consolidata restringere le regole ad una forma normale. Fra le forme normali proposte in letteratura, la più nota è sicuramente la Chomsky Normal Form(CNF) che restringe una regola ad avere come corpo o una simbolo terminale, o una coppia di simboli non-terminali. L'esempio di grammatica dato sopra segue questa forma. Si noti infine come queste restrizioni non riducano l'espressività della grammatica ma hanno il solo scopo di rendere più semplice la comprensione e l'utilizzo delle regole.

Indipendentemente dalla forma impiegata per le regole, la derivazione può seguire due tipi di precedenza: sinistra o destra. In entrambi i casi si inizia comunque con il simbolo speciale iniziale S. Nella derivazione sinistra si espande(o si riscrive), attraverso l'applicazione di una delle regole presenti nella grammatica, sempre la categoria non terminale più a sinistra. L'applicazione di ogni regola produrrà una nuova sequenza di simboli terminali e non, che a loro volta permetteranno l'applicazione di nuove regole. Nel caso di derivazione destra invece si procede in modo del tutto similare ma si espande di volta in volta la categoria più a destra. E' importante sottolineare come nel formalismo finora descritto e sotto la condizione che siano applicate le stesse regole, l'ordine col quale si applicheranno le regole non influenzerà la derivazione. Infine è ormai pratica consolidata rappresentare le derivazioni con strutture ad alberi, detti comunemente *parse tree*.

Dalla formalizzazione della grammatica appena discussa, si sarebbe portati a pensare che una grammatica *context-free* sia una meta-teoria sufficientemente espressiva per rappresentare le grammatiche dei linguaggi naturali. Ciò implicherebbe che tutte le regole sintattiche di ogni linguaggio possano essere rappresentate in questo modo. Tuttavia, sebbene tale teoria rappresenti abbastanza bene la gerarchia dei costituenti, e attraverso la ricorsione riesca a catturare la produttività sintattica dei linguaggi naturali, essa ha evidenziato alcune lacune nel processo di elaborazione di alcuni fenomeni.

1.2 Grammatiche più espressive

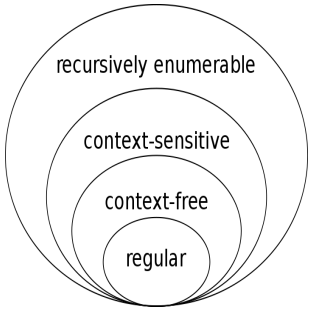


Figura 1.1. Gerarchia delle classi di grammatiche formali.

Nonostante siano pochi, esistono nel linguaggio naturali costrutti che richiedono un'espressività superiore a quella delle grammatiche context-free. Nel 1956 Chomsky definì la sua gerarchia, mostrata in figura 1.1, di classi di grammatiche formali così da relazionare queste ultime in termini di espressività(definita sulla base dei costrutti esprimibili) e complessità. Un anno dopo, Chomsky dimostrò come l'inglese non possa essere descritto da un linguaggio regolare ma non prese nessuna posizione sulla possibilità o meno di analizzarlo con una grammatica context-free. Negli anni successivi si diffuse comunque sempre più la convinzione che i linguaggi naturali richiedessero una maggiore capacità espressiva.

Per avere una rigorosa dimostrazione dell'impossibilità di esprimere alcuni costrutti del linguaggio naturale con grammatiche context-free occorse aspettare trenta anni. Fu Stuart Shieber a dimostrare in modo rigoroso quello che fino ad allora era stata solo una vaga congettura. La base della sua dimostrazione fu una particolare costruzione nello svizzero-tedesco che risulta essere isomorfo ad un noto linguaggio non-context-free. La versione inglese delle espressioni analizzate da Shieber sono simili alle seguenti espressioni:

1. We helped Hans paint the house.
2. We let Hans paint the house.

Con queste espressioni Shieber volle evidenziare la costruzione delle clausole subordinate nello svizzero tedesco. In tali costrutti infatti le parole *Hans* e *house* precedono rispettivamente i verbi *help* e *paint*. Tale costruzione potrebbe, almeno in teoria, giustificare la presenza di un numero imprecisato di espressioni verbali alla fine della frase, a patto che vi siano un numero adeguato di argomenti antecedenti. Rispettare tali vincoli è necessario per avere un'espressione grammaticalmente corretta.

L'analisi di Shieber si rivolse allora ai diversi casi richiesti generalmente dai verbi. Sfruttando il risultato di Hopcroft e Ullman per i linguaggi formali,

e la proprietà dei linguaggi context-free per la quale essi sono chiusi rispetto all'omomorfismo e l'intersezione con i linguaggi regolare, Shieber evidenziò l'impossibilità per i linguaggi context-free di rappresentare i linguaggi naturali.

La dimostrazione di questo enunciato parte dal particolare costrutto con forma $a^n b^m c^n d^m$, conosciuto come dipendenza seriale incrociata, che Hopcroft e Ullman dimostrarono non essere context-free. Shieber definì allora un omomorfismo da applicare ai precedenti costrutti svizzero-tedeschi e successivamente lo intersecò con il linguaggio regolare $wa^*b^*xc^*d^*y$, ciò lo condusse alla definizione del linguaggio $wa^n b^m xc^n d^m y$. Quest'ultimo linguaggio fu la dimostrazione che il linguaggio naturale non può essere espresso da una grammatica context-free.

L'inadeguatezza delle grammatiche context-free spinse la ricerca a concentrarsi su altre classi di linguaggio più espressive. I linguaggi dipendenti dal contesto(*context sensitive*), ad esempio, offrono una enorme capacità espressiva ma ovviamente il prezzo da pagare è in termini di complessità computazionale. Sono in molti tuttavia a credere che tale espressività sia non necessaria per la descrizione della sintassi del linguaggio naturale. Fra questi è possibile citare Joshi che propose nel 1985 le grammatiche debolmente dipendenti dal contesto. Lo scopo di Joshi era quello di definire una nuova classe di linguaggi caratterizzata da una capacità espressiva minimamente necessaria. Per definire quale grammatica (e relativo linguaggio) rientri in questa classe Joshi propose le seguenti proprietà:

1. contengono propriamente tutti i linguaggi context-free.
2. possono essere elaborati in tempo polinomiale.
3. possono rappresentare costrutti complesse, come la dipendenza lineare incrociata.
4. godono della crescita lineare.

Su queste specifiche di classe sono stati col tempo definiti diversi formalismi. La necessità di trattare vincoli linguistici non locali è fra le motivazioni principali dietro lo sviluppo di tali grammatiche. Per esempio le grammatiche basate su aggiunzioni di alberi(tree adjoining grammars) permetto la costruzione di vincoli fra nodi arbitrariamente distanti fra di loro.

Nelle grammatiche basate su aggiunzioni di alberi la costruzione dell'albero sintattico avviene ricorsivamente attraverso l'applicazione di due operazioni di base: aggiunzione e sostituzione. Si noti come queste operazioni, non essendo limitate alla concatenazione di stringhe, estendono la capacità espressiva del formalismo permettendogli di contenere i linguaggi context-free, e alcuni linguaggi del tipo di $\{a^n b^m c^n d^m | n, m \geq 0\}$. Sfortunatamente tale capacità espressiva incrementa la complessità computazionale dell'operazione di analisi(parsing) portandola dal $O(n^3)$ delle grammatiche context-free a $O(n^6)$ per un espressione linguistica di lunghezza n sotto l'assunzione che la grammatica sia costante.

Oltre alle grammatiche basate su aggiunzioni di alberi sono stati sviluppati anche altri formalismi, fra i quali vi sono le grammatiche di combina-

toria delle categorie, e quelle basate sul concetto di testa, denominate *head grammar*. Sebbene questi formalismi siano il risultato di indipendenti linee di ricerca, è importante sottolineare come essi siano equivalenti al formalismo delle grammatiche di aggiunzioni di alberi.

1.3 Grammatica delle categorie e sue estensioni

La presenza di strutture per rappresentare legami fra i diversi costituenti permette di modellare dipendenze non locali presenti soprattutto nei fenomeni di coordinazione ed estrazione. Per evidenziare la differenza nel processo analisi nel seguito si introdurranno le principali idee alla base della grammatica delle categorie, *categorical grammar*, e della sua estensione più famosa conosciuta con il nome di *combinatory categorial grammar*.

Nella sua classica definizione la grammatica divide le sue categorie in atomiche e complesse. Mentre le prime sono le classiche categorie $N, NP, S, etc.$, le seconde sono invece costruite ricorsivamente a partire dalle categorie atomiche attraverso l'applicazione di un funtore che ne indica il tipo e la direzionalità. Si consideri, ad esempio, il verbo transitivo *to buy* che richiede come primo argomento alla sua destra la categoria un NP, e alla sua sinistra una seconda categoria NP. Ciò può essere espresso con la seguente categoria $(S\backslash NP)/NP$.

Nella grammatica delle categorie classica la presenza di due sole regole di *applicazione* rende la grammatica di per se altamente lessicale. Un vantaggio, derivante dall'uso di queste regole direzionali, risiede nella possibilità di definire in modo semplice il legame fra sintassi e semantica. Si può infatti immaginare come ad ogni applicazione di una funzione del piano sintattico si possa associare un'applicazione di qualche funzione sul piano semantico, tipo $\beta - reduction$ nel $\lambda - calculus$. La riduzione a due sole regole per la sintassi ed una per la semantica obbliga la grammatica a prendere una forte connotazione lessicale. In una grammatica di questo tipo tutto deve essere definito in termini di categorie lessicali. Ogni modificatore è definito come argomento di un funtore che restituisce la stessa categoria: N/N o $(S \backslash NP)/(S\backslash NP)$. Sfortunatamente, per ottenere la capacità generativa di grammatiche conformi alla classe debolmente dipendente dal contesto ciò non è sufficiente. La grammatica delle categorie classica richiede quindi ulteriori estensioni.

Costruzioni come dipendenze non vincolate e coordinazione di elementi non costituenti richiedono regole aggiuntive per la loro corretta analisi. In letteratura è possibile trovare diversi estensioni della grammatica delle categorie classica, fra queste la Compositional Category Grammmar è sicuramente la più nota. In questa estensione sono introdotti alcune regole addizionali di composizione riportate in tabella 1.3. Con queste regole è possibile analizzare espressioni come *"The agreement which the fund reached."*, la cui derivazione è mostrata in figura 1.2.

Regola	Sintattica	Semantica
Forward application	$X/Y\ Y \rightarrow X$	$\lambda y[X(y)](y) \rightarrow X(y)$
Backward application	$Y\ X \setminus Y \rightarrow X$	$\lambda y[X(y)](y) \rightarrow X(y)$
Forward composition	$X/Y\ Y/Z \rightarrow X/Z$	$\lambda y[X(y)]\lambda z[Y(z)] \rightarrow \lambda z[X(Y(z))]$
Backward composition	$Y \setminus ZX \setminus Y \rightarrow X \setminus Z$	$\lambda z[X(z)]\lambda y[Y(y)] \rightarrow \lambda z[X(Y(z))]$
Permutation	$X\|Y_1) \ldots \|Y_n \rightarrow X\|Y_n) \ldots \|Y_1$	$\lambda y_n, \ldots, y_1[X(y_1, \ldots, y_n)] \rightarrow \lambda y_1, y_n, \ldots [X(y_1, \ldots, y_n)]$
Type raising a	$(X \rightarrow T/(T \setminus X))$	$a \rightarrow \lambda T[Ta]$
Type raising b	$(X \rightarrow T \setminus (T/X))$	$a \rightarrow \lambda T[Ta]$

Tabella 1.1. Schema delle regole per Compositional Category Grammmar.

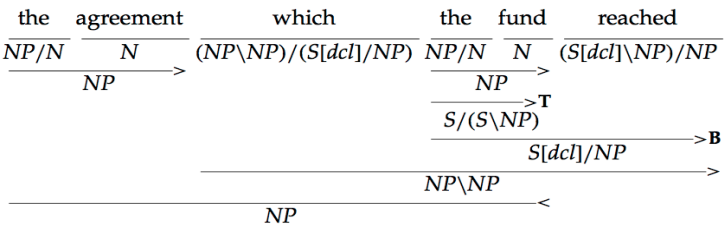


Figura 1.2. Esempio di derivazione per Compositional Category Grammmar..

L'espressione *the fund* ha inizialmente la categoria NP, dopo l'applicazione della regola *Type raising* essa diventa un funtore che attende una frase verbale alla sua destra $S/(S \setminus NP)$. Poiché *reached* è etichettata con la categoria $S[dcl] \setminus NP$[1], le due espressioni *the fund* e *reached* possono allora combinarsi attraverso l'applicazione della regola di *forward composition*. Quest'ultima genererà la categoria $S[dcl]/NP$ che risulta essere la categoria richiesta alla destra del pronome relativo oggetto *which*. Nella teoria delle Compositional Category Grammmar esistono anche ulteriori regole come per esempio la composizione incrociata all'indietro o la coordinazione fra categorie dello stesso tipo per produrre un'ulteriore categoria sempre dello stesso tipo. Tuttavia, nonostante il forte interesse iniziale e il loro utilizzo in numerosi progetti, queste grammatiche vivono ultimamente una marginalizzazione nel campo dell'analisi sintattica automatica. Il motivo di ciò è da imputare principalmente alla prepotentemente diffusione dell'approccio fondato sulla tradizione delle grammatiche delle dipendenze.

1.4 Grammatica delle dipendenze

Le grammatiche basate su dipendenze hanno catturato l'interesse di sempre più ricercatori negli ultimi anni. Il motivo di tale attenzione può essere sintetizzato in tre aspetti principali: semplicità, flessibilità e velocità di analisi.

In molte applicazioni che utilizzano il linguaggio naturale, come ad esempio sistemi di traduzione e di estrazione della conoscenza, questo tipo di rappresentazione semplice e intuitivo ha ampiamente dimostrato la sua utilità nel

[1] La categoria $S[dcl]$ definisce le espressioni dichiarative.

descrivere strutture predicato-argomento. Se comparato con le grammatiche discusse precedentemente, quest'approccio ha poi mostrato una maggiore flessibilità nel modellare linguaggi in cui l'ordine delle parole non sia strettamente vincolato. Infine, ma forse determinante, è lo stretto legame che quest'approccio ha sviluppato con tecniche di apprendimento automatico. Grazie alla sempre più diffusa disponibilità di testi annotati, molta ricerca è stata rivolta su modelli probabilistici addestrati in modo supervisionato che potessero catturare le dipendenze fra le parole. Questo ha permesso ad alcuni parser di raggiungere livelli di correttezza che mediamente si aggirano intorno al 95%. Come già avvenuto nella sezione precedente, questa sezione tenterà di fornire una breve descrizione delle nozioni di base della teoria, tralasciando i dettagli dell'analisi linguistica e focalizzandosi invece su ciò che è strettamente necessario ai fine del corrente lavoro.

Anche se la grammatica delle dipendenze ha una lunga tradizione nella moderna teoria linguistica, è diventato ormai uso comune riferirla al lavoro postumo di Lucien Tesnière pubblicato alla fine degli anni 50. Nelle sue varie sfumature questa grammatica ha comunque come principio cardine la definizione di un legame binario e asimmetrico fra le parole, detta relazione di dipendenza, per generare le strutture sintattiche. Tale relazione lega fra di loro parole sintatticamente subordinate assegnando loro i ruoli di dipendente e *testa*[2]. Come si può vedere in figura 1.3, l'analisi di una semplice espressione è composta da un insieme di relazioni, rappresentate come frecce orientate dalla testa al dipendente ed etichettata con una specifica *relazione* di dipendenza. Il risultato prodotto dall'analisi sintattica può essere quindi formalizzato come un grafo diretto i cui archi sono tutti etichettati con una relazione di dipendenza. Nel grafo ogni nodo corrisponde ad una parola, e si assume l'esistenza di un ulteriore nodo speciale $ROOT$ senza archi incidenti.

Questo tipo di analisi potrebbe apparire come diametralmente opposta rispetto l'analisi basata sulle strutture di frase; tuttavia questa considerazione è per sua natura solo una grossolana semplificazione. Infatti, sebbene l'informazione rappresentata dalla grammatica delle dipendenze sia differente da quella descritta precedentemente, tale differenza riguarda solo il "*che cosa*" è esplicitamente rappresentato. Se nel caso della grammatica delle dipendenze si descrivono relazioni binarie fra coppie di parole(classificate da categorie funzionali come SBJ o OBJ), nel caso della definizione di strutture di frase si indica invece il raggruppamento di parole all'interno della frase in categorie strutturali(p.e. NP e VP). Tuttavia, è bene notare come le frasi possano essere interpretate all'interno del paradigma delle dipendenze permettendo ad ogni parola di rappresentare se stessa e l'intera espressione che dipende da essa. Lo stesso può esser fatto per le relazioni funzionali trattate come configurazioni strutturali. Quindi posto che le dipendenze riescano a catturare elementi es-

[2] La letteratura è piena di termini alternativi per entrambe le categorie. Per il dipendente sono comuni alternative i termini di modificatore o figlio, mentre la testa è spesso indicata coi termini di governante, reggente o genitore.

senziali del linguaggio, occorre definire qualche criterio per i tipi di relazione e distinguere poi fra dipendente e testa.

I criteri per definire le relazioni di dipendenza risiedono in un piano intermedio fra sintassi e semantica. Di seguito sono riportati alcuni dei criteri più comuni per un certo contesto c, testa t, e dipendente d:

1. t determina la categoria sintattico-semantica di c.
2. d fornisce specificazioni semantiche.
3. t è obbligatoria, mentre d no.
4. t selezionando d ne determina lo status: obbligatorio o opzionale.
5. La forma di d dipende da t.
6. La posizione di d è espressa con riferimento ad t.

Ovviamente questa lista non è esaustiva, ne tanto meno esaurisce le questione relative alla definizione dei criteri. Capita tuttavia che a volte vi sia un'inclinazione per asserire come diversi livelli di dipendenze siano necessari: livelli per descrivere fenomeni come la morfologia, la sintassi o la semantica. Attualmente comunque è pratica comune far risiedere tutte le relazioni di dipendenza della grammatica in unico livello. Discorso a parte va comunque riservato per i tipi di composizione endocentrici e esocentrici.

La composizione[3] è un procedimento morfologico che permette di formare parole nuove, o neologismi, attraverso la combinazione di due (o più) morfemi lessicali. I composti in cui è possibile identificare un elemento con funzione di testa sono detti endocentrici, mentre i composti come pellerossa (indigeno dell'America del Nord) o casco blu (militare dell'ONU), in cui cioè nessuno dei due elementi contribuisce in modo prevalente a determinare le caratteristiche semantico-sintattiche del composto sono detti esocentrici. La distinzione fra questi due tipi di composizione è anche assimilabile alla caratterizzazione dei fenomeni fra una testa e il suo complemento e fra una testa e un suo modificatore. Nello specifico i primi rientrano nei composti esocentrici mentre i secondi in quelli endocentrici.

Infine, è importante sottolineare come la grammatica delle dipendenze non sia immune da questioni irrisolte o situazioni poco chiare. Nei casi di articoli, verbi ausiliari o strutture di frasi preposizionali l'approccio basato sulle dipendenze non offre un'unica condivisa soluzione; manca infatti una posizione largamente accettata in merito alla possibilità di analizzare questi aspetti come relazioni di dipendenza. In alcune teorie si avanza la proposta di trattare i verbi ausiliari come testa e il corrispondente verbo come dipendente, ma ciò ha

[3] Semanticamente un composto è prodotto dai significati dei due costituenti. I due principali tipi di relazione sono determinazione e coordinazione. Nel primo caso un costituente esprime la parte più generale e importante del composto, mentre l'altro ne specifica le caratteristiche: ad es., una cassaforte è una cassa particolarmente forte. Nella coordinazione invece, i due costituenti contribuiscono equamente al significato del composto: ad es., una cassapanca è sia una cassa su cui ci si può sedere sia una panca con uno scomparto in cui riporre qualcosa.

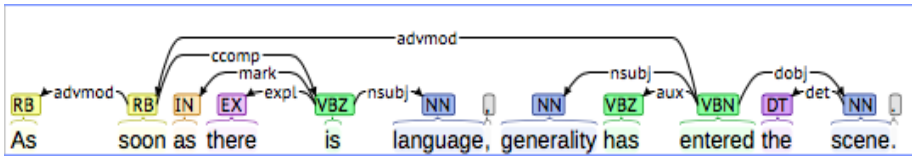


Figura 1.3. Esempio di analisi per la frase "As soon as there is language, generality has entered the scene.". Gli archi rappresentano le dipendenze fra le parole, e le loro etichette il tipo. I quadratini colorati invece sono le categorie sintattiche assegnate alle parole secondo una grammatica basata su strutture di frasi.

ricevuto critiche concernenti la diversa natura che tali dipendenze imporrebbero sul costrutto. Anche il fenomeno della coordinazione è fonte di problemi. Come accennato sopra, la visione strutturalistica della coordinazione la riconduce ad una composizione endocentrica. Posto che questa composizione possa avere più teste capaci di sostituire sintatticamente l'intero costrutto, risulta quindi non immediato, o addirittura inappropriato, modellare questa situazione in termini di relazione binaria. Nell'analizzare, per esempio, le seguenti espressioni:

1. James likes movies and books.
2. She is running and singing.

appare evidente come nella prima frase sia movies sia books possano essere dei candidati a ricoprire il ruolo di testa. Tuttavia non esistono criteri che guidino la scelta in favore di uno o dell'altro, in quanto entrambi ricoprono la funzione di oggetto diretto. Anche nella seconda frase la coordinazione fra i due verbi pone lo stesso problema. Attualmente, è prassi comune trattare la parte sinistra di una congiunzione come la testa, e assegnare alla destra il ruolo di dipendente. Benché tale approccio può trovare dei fondamenti in semantica, è importante sottolineare come esso sia solo una tendenza nel trattare la coordinazione.

Nonostante le questioni appena descritte, l'analisi che utilizza una grammatica delle dipendenze è diventata negli ultimi anni molto popolare. Grazie soprattutto agli avanzamenti tecnologici e alla disponibilità di una sempre crescente quantità di dati, l'elaborazione automatica ha ormai raggiunto livelli di prestazione e affidabilità così alti da permettere di ridefinire le soglie per la definizione dello stato-dell-arte. Tuttavia è importante sottolineare come col tempo si sia osservato una proliferazione di proposte di nuove relazioni di dipendenza. Solitamente ciò accade a causa delle necessità di rappresentare qualche particolare costrutto in una determinata lingua sotto esame. Solo ultimamente però gli sforzi si sono concentrati nel tentativo di riunire sotto un'unica teoria un insieme di dipendenze fondamentali. Questi sforzi sono all'origine delle dipendenze universali.

1.5 Dipendenze universali

Le dipendenze universali (UD) sono un tentativo promosso da diversi gruppi di ricercatori per creare una collezione di testi, scritti in diverse lingue e annotati secondo il sistema proposto dalla teoria delle dipendenze. L'obiettivo principale di UD è definire un'unica e coerente metodologia per rappresentare le dipendenze. L'annotazione sintattica che ne deriva è ancora espressa in termini di dipendenze e conserva ancora le parole come unità di base per l'analisi grammaticale, ma per ottenere una analisi coerente su diversi linguaggi UD usa un insieme di categorie, funzionali e strutturali, condivise da tutti i linguaggi.

Gli aspetti morfologici dell'analisi di una parola sono suddivisi su tre livelli: lemma, parti della frase e un insieme di caratteristiche per le proprietà lessicali e grammaticali associate al determinato termine. La tabella 1.5 elenca l'insieme delle etichette per le parti della frase(part-of-speech) condiviso fra i vari linguaggi. Quest'insieme di categorie morfologiche è basato sul sistema *Interset*, e ad ogni categoria è associato un insieme di valori: ad esempio *Number* ha come valori {*singular,plural,dual,plurale tantum*} e *collective*.

Parole di classe aperta	Parole di classe chiusa	Altre
ADJ adjective	ADP preposition/postposition	PUNCT punctuation
ADV adverb	AUX auxiliary	SYM symbol
INTJ interjection	CONJ coordinating conjunction	X unspecified POS
NOUN noun	DET determiner	
PROPN proper noun	NUM numeral	
VERB verb	PART particle	
	PRON pronoun	
	SCONJ subordinating conjunction	

Tabella 1.2. Part-of-speech tags in UD.

Le *relazioni grammaticali* organizzano le relazioni fra tre tipi di strutture: nominali, clausole e modificatori di parole. Sebbene esiste una velata distinzione fra dipendenti più o meno importanti, tutto è trattato in termini di aggiunzioni e non vi è nessun tentativo di distinguerle dagli argomenti.

UD specifica di estendere le tipologie di dipendenze evidenziando se i dipendenti siano frasi o clausole: come ad esempio *nsubj* e *csubj*, *dobj* e *ccomp*, o *advmod* e *advcl*. Essa specifica inoltre di distinguere fra clausole indipendenti dal complemento, cioè fra quelle che possiedono un soggetto interno e quelle che invece necessitano un *controllo* obbligatorio da parte del soggetto del predicato governante, *ccomp* e *xcomp*. Le restanti clausole dipendenti sono tutte trattate come *modificatori*. UD richiede di differenziare le appendici ai predicati da quelle ai nominali: una clausola avverbiale *advcl* modifica un predicato mentre *acl* un nominale.

UD infine distingue la combinazione dalla modificazione o complementazione utilizzando tre relazioni: *mwe*, *name*, *compound*. La prima è usata per espressioni grammaticali fisse con parole di funzione, come ad esempio "*instead of*" espressa con $mwe(instead, of)$. Mentre *name* è utilizzato per nomi costituiti da più nomi propri, cioè espressioni come "*The Lord of the Rings*".Altri tipi di espressioni multi-parola che risultano essere ricorrenti nella lingua, come nomi composti ma non solo, sono etichettati con la relazione *compound*.

Dipendenze principali dei predicati di clausola			
Dip nominale	Dip di predicato		
nsubj	csubj		
nsubjpass	csubjpass		
dobj	ccomp		xcomp
iobj			
Dipendenze secondarie dei predicati di clausola			
Dip nominale	Dip di predicato		Modificatore di parole
nmod	advcl		advmod
			neg
Dipendenti speciali di clausola			
Dip nominale	Ausiliari	Altro	
vocative	aux	mark	
discursive	auxpass	punct	
expl	cop		
Dipendenti del nome			
Dip nominale	Dip di predicato		Modificatore di parole
nummod	acl		amod
appos			det
nmod			neg
Marcatori di caso, preposizioni, possessivi			
case			
Coordinazione			
conj	cc	punct	
Compounding and unanalyzed			
compound	mwe	goeswith	
name	foreign		
Relazioni di giunzione			
list	parataxis	remnant	
dislocated		reparandum	
Altro			
Testa di frase	Dipendenze non-specificate		
root	dep		

Tabella 1.3. Relazioni di dipendenza in UD.

Parole di contenuto

Ogni parola può dipendere o da un'altra parola o dall'entità speciale *ROOT*. Le parole di contenuto sono parole alle quali è associato un significato e sono

collegate attraverso relazione di dipendenza, mentre le parole che aggiungo informazione strutturale per la grammatica sono semplicemente attaccate alle parole di contenuto. Sulla base di questi principi, il trattamento della copula e degli ausiliari prevede che essi non siano la testa di una clausola, ma dipendano invece dal predicato lessicale.

La coordinazione segue invece un trattamento per il quale l'espressione più a sinistra diviene la testa, mentre le altre congiunzioni coordinanti dipendono da essa. Ogni elemento, nei diversi casi di proposizioni, post-condizioni o clitico, è trattato come dipendente del nome a cui è attaccato o che introduce. Ad esempio la relazione *nmod* esprime una certa forma di reazione obliqua o di aggiunzione che può essere ulteriormente specificato dal caso o dagli aspetti morfologici.

Benché la scelta di fondare la rappresentazione sintattica sulle parole di contenuto possa sembra in contro tendenza rispetto ad altre teorie sintattiche che rivolgono invece l'attenzione verso un'analisi funzionale dei termini, è importante ricordare che tale scelta sia il frutto della necessità di rendere il più possibile omogeneo l'analisi sintattica fra linguaggi differenti. Infatti assegnando una priorità di combinazione alle parole di contenuto si aumenta la possibilità di trovare parallelismi nelle strutture fra linguaggi diversi. In UD i termini funzionali diventato delle appendici delle parole di contenuto che ne specificano le caratteristiche.

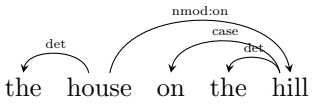
1.5.1 Dipendenze universali arricchite

Lo scopo di arricchire le dipendenze è quello di esplicitare attraverso una certa marcatura soggetti esterni, ruoli esterni nella clausole relative o propagare relazioni oltre le congiunzioni. Il motivo di ciò riguarda l'esperienza fatta su alcuni sistemi per la comprensione *superficiale* del linguaggio. Tali sistemi prediligevano utilizzare come relazioni fra le parole di contenuto una rappresentazione alternativa, denominate *collapsed* o *CCprocessed*. Tale rappresentazione, sebbene da un lato rendeva l'albero delle dipendenze un grafo, dall'altro esplicitava relazioni altrimenti implicite, come accade ad esempio nella frase *"John starts to read"*. Qui la classica rappresentazione delle dipendenze perde la relazione diretta fra il controllore *John* e il verbo controllato *read*, ma all'interno della rappresentazione CCprocessed è invece creato un arco addizionale per il soggetto.

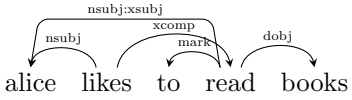
In UD manca un adeguato trattamento di questa rappresentazione alternativa. L'obiettivo delle UD arricchite è quello di estendere le relazione di base di UD con le seguenti relazioni addizionali.

- **Modificatori arricchiti** : secondo le specifiche di UD le teste delle frasi preposizionali sono i complementi e non le preposizioni. Nella versione collassata delle dipendenze standard, tuttavia, tutti i modificatori nominali includono la preposizione nei loro nomi di relazione. Lo stesso accede anche nel caso di frasi preposizionali più complesse come i modificatori con

clausola avverbiale (*advcl*) o aggettivale (*acl*).

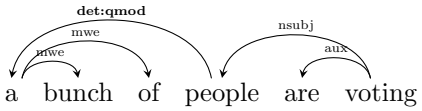


- **Congiunzioni arricchite** : le relazioni di congiunzioni sono aumentate con la loro congiunzione coordinante.
- **Governatori e dipendenti propagati** : le relazioni implicite sono esplicitate attraverso l'utilizzo di relazioni aggiuntive. Ciò accade, ad esempio, nei casi di frasi nominali congiunte o espressioni verbali congiunte.
- **Il soggetto dei verbi controllati** : nelle specifiche di UD non esiste una relazione fra un verbo controllato ed il suo controllore, o soggetto. Per questo motivo tale relazione è qui introdotta con la forma di $nsubj : xsubj$.

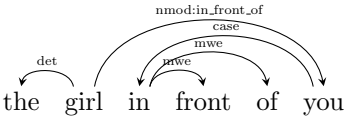


Le rappresentazioni appena presentate forniscono un utile contributo per l'analisi di molte frasi della lingua inglese. Tuttavia l'analisi condotta da Schuster e Manning ha evidenziato come esse non riescano ad esplicitare direttamente alcuni costrutti. Fra questi costrutti gli autori dell'analisi si focalizzano soprattutto sulle frasi partitive nominali e quelle con preposizioni multi-parola. Schuster e Manning propongono quindi di eliminare alcune relazioni dalla definizione originaria di UD e di aggiungerne delle nuove per rappresentare meglio questi costrutti.

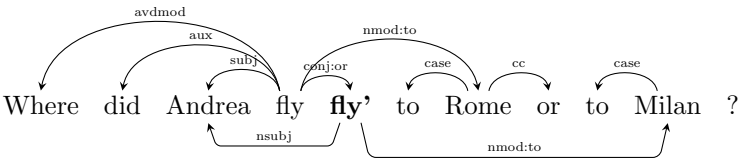
- **Costruzioni partitive e di *light noun*** : nelle espression come "*both of the kids*" la frase nominale principale è trattata semanticamente come la testa della frase, mentre il suo determinatore quantificante come semplice espressione multi-parola. Tali determinatori quantificanti sono etichettati con *det* : *qmod*. Nelle costruzioni denominate *light noun*, come ad esempio "*a bunch of people*", la frase saliente si manifesta tipicamente nella seconda frase nominale ma nell'analisi classica la testa è tipicamente assegnata a *light noun*, in questo caso *bunch*. La proposta avanzata dall'approccio delle UD arricchite è quella di analizzare il costrutto come nel caso precedente:



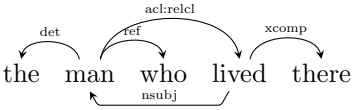
- **Preposizioni multi-parole** : le preposizioni multi-parole, come *in front of*, possono nascondere la reale relazione che sussiste fra due parole di contenuto. Per questo motivo le espressioni multi-parole sono trattate come forme piatte e le parole di contenuto sono direttamente collegate:



- **Preposizioni di congiunzione e frasi preposizionali** : alcune clausole contenenti preposizioni congiunte sono difficili da analizzare. Espressioni come "*Where did Andrea fly to Rome or to Milan ?*" richiedono, per una più comprensiva analisi, la duplicazione di qualche nodo come proposto dall'approccio CCprocessed. Risulta infatti molto utile rappresentare nel grafo la relazione *nmod* : *to* sia fra *fly* e *Rome* sia fra *fly* e *Milan*. Per questo motivo si duplica il nodo del verbo *to fly* e lo si collega al nodo originario con la relazione $conj$: or:



- **Pronomi relativi** : come accade nella rappresentazione collassata delle dipendenze classiche, i riferimenti dei pronomi sono direttamente collegati con il loro governante. In aggiunta si inserisce un'ulteriore dipendenza, ref, fra il pronome ed il suo riferimento:



1.6 Rete neurale per l'analisi delle strutture di dipendenze

1.6.1 Preliminari

Il problema di analizzare automaticamente un'espressione $S = w_0 w_1 \dots w_n$ può essere definito in termini di costruzione di un grafo di dipendenze G da associare all'espressione. Le tecniche più utilizzate per tale costruzione possono essere distinte in due categorie: approcci basati sui dati o sulle grammatiche. Nel primo rientrano la maggior parte degli approcci derivanti dall'apprendimento automatico, mentre il secondo si basa sulla definizione formale di una grammatica che trasforma il problema nella verifica di appartenenza dell'espressione alla grammatica data.

Come visto nelle sezioni precedenti le grammatiche formali hanno il compito di restringere l'insieme delle stringhe alle sole licenziate dalla grammatica.

Nel contesto dell'analisi delle dipendenze gli approcci utilizzati non si discostano molto dalle precedenti tecniche di analisi. L'analisi delle dipendenze può infatti essere modellato come un problema di soddisfacimento vincolato. La grammatica definisce i vincoli sulle strutture e l'analisi ha il compito di trovare il grafo che soddisfa il "maggior numero" di vincoli imposti dalla grammatica. Di solito però tali vincoli possono risultare troppo rigidi, da qui l'idea di rilassare leggermente l'approccio e permettere la violazione di alcuni vincoli. Associando infine dei pesi ai vincoli presenti l'analisi si può ridurre il problema alla costruzione del grafo con il miglior punteggio.

Nell'approccio supervisionato, invece, il problema originario è suddiviso in due sotto-problemi: apprendimento del modello e suo utilizzo per l'analisi. Generalmente per l'apprendimento dei modelli si utilizza un insieme di frasi annotate con grafi di dipendenze. Sebbene il processo d'annotazione è più semplice rispetto ad altri formalismi grammaticali, è pratica comune addestrare il modello su collezioni di testo precedentemente annotate. In questo modo si facilita anche la comparazione fra modelli prodotti con tecniche diverse. L'utilizzo del modello infine è abbastanza immediato, il modello infatti restituirà la più probabile fra le analisi possibili.

L'approccio supervisionato per l'analisi delle dipendenze, appena descritto, ha visto nell'ultimo decennio l'affermarsi principalmente di due metodi: uno basato sulla teorie dei grafi, e l'altro su transizione di stato. Nel primo metodo il modello indotto ha il compito di assegnare un punteggio ai possibili grafi appartenenti allo spazio dei grafi candidati, mentre per l'analisi si applicano algoritmi per il calcolo del *maximum spanning tree* che equivale a trovare il grafo col più alto valore di probabilità. Nel secondo metodo invece si definisce una macchina a stati e si modella il problema come il problema di predire la prossima transizione data una certa storia di transizioni.

I sistemi basati su transizione di stato predicono una sequenza a partire da una configurazione iniziale fino ad uno stato terminale che produce l'albero delle dipendenze. Nell'approccio proposto da Chen e Manning, descritto nel seguito, si usa una tecnica *golosa* che utilizza un classificatore per predire la prossima transizione in base alle caratteristiche estratte dalla configurazione corrente. Tuttavia, per via delle propagazione degli errori tali tecniche tendono in generale a comportarsi peggio nell'elaborazione del testo rispetto a tecniche basate su ricerca e *backtracking*. Ciononostante i risultati presentati da Chen e Manning hanno mostrato un'accuratezza comparabile.

Il sistema proposto segue l'approccio classico e definisce una configurazione c come composta da uno *stack*, un *buffer*, e l'insieme di dipendenze finora risolte. Data una espressione da analizzare $s = w_1 \dots w_n$, la configurazione iniziale è definita come $c = \langle [ROOT], [w_1 \dots w_n], \emptyset \rangle$. Mentre la configurazione finale si raggiunge nel momento in cui il buffer non contiene più nessuna parola.

Le operazioni che definisco una transizione di stato sono di tre tipi:

- *LEFT-ARC(d)* : aggiunge la dipendenza d alla configurazione fra il primo elemento dello stack verso il secondo e rimuove quest'ultimo. Applicata sotto la condizione che la capienza dello stack sia ≥ 2.
- *RIGHT-ARC(l)* : aggiunge la dipendenza d alla configurazione fra il secondo elemento dello stack verso il primo e rimuove quest'ultimo. Applicata sotto la condizione che la capienza dello stack sia ≥ 2.
- *SHIFT* : muove il prossimo elemento del buffer, se esiste, nello stack.

Tradizionalmente le informazioni accessibili per predire la corretta transizione riguardano le etichetta di part-of-speech, le informazioni sui governanti e la loro posizione all'interno dello stack o del buffer. Tuttavia queste informazioni soffrono di alcune problematiche come incompletezza, sparsità, e carico computazionale. Se la questione dell'incompletezza è di difficile risoluzione, gli ultimi due problemi possono essere trattati in qualche modo.

Il problema della sparsità riguarda principalmente le caratteristiche lessicali. Queste ultime risultano fondamentali per la corretta assegnazione delle dipendenze. Da qui la questione computazionale che è legata alla necessità di estrarre tali caratteristiche per poter predire la transizione.

E' infatti importante sottolineare come fino a qualche anno fa i sistemi impiegavano la maggior parte del tempo d'analisi nel processo di estrazione delle caratteristiche linguistiche. Ciò accadeva poiché le parole erano rappresentate da vettori di grandi dimensioni. Solo recentemente con l'avvento di rappresentazioni caratterizzate da vettori di dimensioni più contenute si è potuto concentrare l'attenzione sulle scelte da fare, come avviene nell'approccio proposto da Chen e Manning.

1.6.2 Parser basato su reti neurali

Nel 2014 Chen e Manning proposero di utilizzare un classificatore basato su reti neurali per effettuare le scelte durante la fase di decisione delle transazioni. La rete, progettata per apprendere una compatta e densa rappresentazione di parole, part-of-speech, e etichette di dipendenza, risultò molto veloce e affidabile nonostante l'esiguo numero di caratteristiche utilizzate(soltanto 200)[4].

Per ogni parola, etichetta POS e dipendenza si utilizza un vettore di dimensione d: $e_i^w \in \mathbb{R}^d$, $e_i^{pos} \in \mathbb{R}^d$, e $e_i^{dep} \in \mathbb{R}^d$ rispettivamente. Questi vettori una volta riuniti in forma matriciale danno luogo alle matrici $E^w, E^{pos}, E^{dep} \in \mathbb{R}^{d \times N}$ dove N è la grandezza del dizionario.

Ai fini della predizione sono costruiti di volta in volta gli insiemi S^w, S^{pos}, S^{dep}, i cui elementi dipendono dalla configurazione corrente dello stack e del buffer. Chen e Manning propongono una rete neurale classica, rappresentata in figura 1.4, con un solo livello intermedio e con dati d'ingresso costruiti a partire dai vettori degli elementi contenuti in S^w, S^{pos}, S^{dep}. La funzione d'attivazione per il livello di ingresso è una funzione cubica:

[4] Il numero delle dimensioni impiegate è stato scelto sperimentalmente.

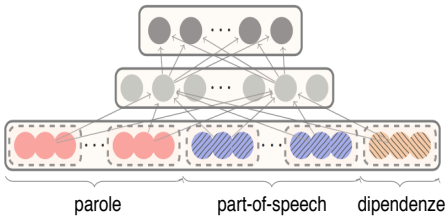


Figura 1.4. Architettura della rete neurale per l'analisi delle dipendenze.

$$h = (W_1^w x^w + W_1^{pos} x^{pos} + W_1^{dep} x^{dep} + b_1)^3 \tag{1.1}$$

dove $W^w \in \mathbb{R}^{d_h \times (d \cdot n_w)}$, $W^w \in \mathbb{R}^{d_h \times (d \cdot n_{pos})}$,$W^w \in \mathbb{R}^{d_h \times (d \cdot n_{dep})}$, $b_1 \in \mathbb{R}^{d_h}$ con n_w, n_{pos}, n_{dep} la cardinalità degli elementi scelti e $x^w = [e^w_{w1}, e^w_{w2}, \ldots, e^w_{wn}]$ derivato da $S^w = \{w_1, \ldots, w_n\}$. I vettori x^{pos} e x^{dep} sono costruiti in modo analogo.

Usando una funzione $f(x) = x^3$ è possibile modellare i termini del prodotto di $x_i x_j x_k$ per ogni elemento in ingresso. Questa funzione d'attivazione è rivolta quindi a catturare l'interazione di tre elementi. Tale scelta, rispetto alle più classiche funzioni *tanh* o *sigmoid*, segue le convinzioni degli autori che attribuiscono a questa funzione una desiderabile utilità nell'analisi delle dipendenze. Infine per ottenere la probabilità multi-classe si utilizza una classificatore $p = softmax(W_2 h)$, con $W_2 \in \mathbb{R}^{|\tau| \times d_h}$, all'uscita del livello intermedio.

La fase di addestramento è svolta generando esempi $\{(c_i, t_i)\}_{i=1}^m$, dove c_i è la configurazione corrente e $t_i \in \tau$ è la transizione effettuata dall'oracolo. Ogni esempio è corredato da dipendenze già annotate usando un oracolo *shortest-stack* che antepone l'applicazione dell'operatore $LEFT-ARC_l$ a $SHIFT$. Su questi esempi, utilizzando la tecnica *AdaGrad* su mini-*batch* e infine applicando la tecnica del *dropout*, si minimizza il valore della funzione caratterizzante l'errore definita come:

$$L(\theta) = -\sum_i log p_{t_i} + \frac{\lambda}{2} ||\theta||^2 \tag{1.2}$$

dove θ è l'insieme di tutti i parametri del modello e l'ultimo termine è un fattore di regolarizzazione l_2.

Infine la fase di analisi prevede ad ogni passo l'estrazione dalla configurazione corrente dei vettori relativi alle parole, alle annotazioni POS, e alle dipendenze. Per ogni transizione si calcola il valore attraverso una propagazione dell'ingresso sulla rete e si sceglie la transizione con il valore maggiore:

$$t = arg\ max_t(W_2(t, \cdot)h(c)) \tag{1.3}$$

Essenzialmente il parser opera molte moltiplicazioni matriciali invece delle più costose operazioni di congiunzioni di caratteristiche e di ricerca delle stesse all'interno di enormi tabelle, operazioni queste tipiche dei precedenti approcci.

1.7 Sommario e riferimenti per approfondimenti

Gli argomenti trattati in questo capitolo sono abbastanza generali e trovano facilmente posto all'interno di ogni testo introduttivo l'elaborazione del linguaggio naturale. Le grammatiche senza contesto e i loro linguaggi sono introdotti molto bene nel quarto e sesto capitolo in [65], ma anche in [118]. La gerarchia per le diverse tipologie di linguaggi è stata introdotta da Chomsky in [23, 24], mentre la discussione riguardante la posizione del linguaggio naturale non è riferibile ad un singolo lavoro bensì è sviluppata in una serie di lavori; tuttavia fra questi è sicuramente degno di menzione il lavoro di Pullum e Gazdar [124]. L'incapacità delle grammatiche senza contesto di contenere il linguaggio naturale è invece discussa in[14, 137] e [96].

Due paradigmi non trattati in questo capitolo ma molto diffusi sono TAG e HPSG. Le grammatiche basate su aggiunzione di alberi sono inizialmente presentate in[71], e sono poi trattate in più dettaglio nei lavori successivi[70, 136]. La definizione di un algoritmo polinomiale per l'analisi basata su TAG è presentato invece in [145] e in [131]. HPSG è invece ampiamente descritto in [121]. Questo paradigma è il risultato dell'integrazione delle idee provenienti da diversi approcci come Generalized Phrase Structure Grammar [52], Lexical-Functional Grammar [29] e la teoria del "Government and Binding" [25].

Una buona introduzione alla grammatica delle categorie e alle sue generalizzazioni può essere trovato in [148]. Nei lavori di Steedman e in particolare in [140] è invece possibile trovare una versione lessicalizzata della grammatica orientata ai tipi. Infine nel lavoro di Clark e Curran[28] è possibile trovare una descrizione di un efficiente algoritmo per eseguire l'analisi sintattica utilizzando questa grammatica.

La grammatica delle dipendenze ha la sua origine nel lavoro di Tesnière [142]. Ma anche altre teorie hanno contribuito alla sua formazione: fra queste la Functional Generative Description[135], la Dependency Unification Grammar[62] e la Word Grammar[66]. Altri approcci per la definizione di formalismi per la grammatica delle dipendenze sono invece basati sul soddisfacimento di vincoli. In questa categoria vi sono la Weighted Constraint Dependency Grammar[132], la Functional Dependency Grammar[68], ed la Extensible Dependency Grammar[33]. Le questioni non ancora risolte riferite alla grammatica delle dipendenze sono ampiamente trattate in [116]. Infine per approfondire la discussione sulle dipendenze universali un buon punto di partenza è il lavoro di McDonald [100], ma informazioni più aggiornate sono disponibili sul sito del progetto[5]. Le dipendenze universali arricchite sono invece discusse in un recente lavoro di Schuster e Manning [133].

La rapida introduzione sulla problematica del *parsing* basato sulla grammatica delle dipendenze può essere approfondita nel terzo capito di [116]. L'elaborazione vincolata per l'analisi delle dipendenze fu introdotta da Maruyama in [98], mentre la definizione di un modello orientato ai dati per il

[5] http://universaldependencies.org

parsing fu descritto da Eisner in [36]. La definizione dei metodi basati sui grafi o sulle transizione sono invece riconducibili ai lavori di Buchholz e Marsi in [16] e nel lavoro di McDonald e Nivre [99]. Infine l'approccio di Chen e Manning basato sull'impiego di una rete neurale è descritto in [22].

2

La questione semantica nel linguaggio naturale

Definire il significato di una parola, o di un'espressione, è uno degli aspetti più intricati all'interno della linguistica tradizionale. Pur assumendo l'esistenza di una qualche struttura non è infatti semplice definire come assegnarle un'interpretazione. La presenza nel linguaggio naturale di fenomeni di ambiguità, riferimenti anaforici o ellittici e presupposizioni, complica notevolmente l'analisi semantica. All'interno della semantica computazionale queste difficoltà sono poi ulteriormente aggravate dalla necessità di legare una descrizione del significato prevalentemente astratta ad una formalizzazione logica.

Questo capitolo discute i principi sviluppati nei diversi approcci alla semantica nel corso degli ultimi due secoli. Si inizia con le diverse caratterizzazioni della semantica, dapprima dalla prospettiva linguistica e successivamente dalla relativa controparte filosofica. È allora proposta la teoria definita da Montague e alcune sue estensioni per rappresentare il discorso. Infine, dopo un piccolo accenno al trattamento degli eventi proposto da Davidson, sarà discussa la teoria di rappresentazione astratta del significato.

2.1 Semantica nella tradizione delle strutture sintattiche

La nozione della semantica assume diverse connotazioni in base alla prospettiva dalla quale si analizza. Una prima distinzione può essere fatta sugli obiettivi dietro l'indagine semantica. Se per i filosofi del linguaggio, ad esempio, gli scopi dell'indagine risiedono nel valore di *verità* da associare ad un espressione, nella composizione di un significato a partire da elementi costitutivi manifesti, e nella questione ontologica legata al suo riferimento col mondo esterno. Per i linguisti ultra moderni (da Chomsky in poi) invece l'interesse è maggiormente concentrato sulle competenze di chi *parla*: ovvero sul tipo di rappresentazione gli interlocutori possiedono, come essa sia costituita e in quali termini si sviluppa il legame fra il piano sintattico e semantico del linguaggio.

In linguistica i primi studi sulla semantica possono essere ricondotti a Bréal. Nel suo *Essai de sémantique*, Bréal analizzò ciò che oggigiorno è in-

dicato con il termine di semantica diacronica: cioè un esame sistematico del mutamento del significato delle parole, ovvero, del fenomeno dell'evoluzione dei sensi delle parole. E solo dopo l'approccio strutturalistico di Saussure nacque la semantica sincronica rivolta invece alle relazione fra i significati. Passando attraverso i campi semantici di Trier e le unità minime di significa di Hjelmslev, si arriva poi a nuove prospettive sugli studi semantici attraverso lo sviluppo delle grammatiche generativo-trasformazionale.

Benché Chomsky fosse abbastanza scettico sulla possibilità di avere una teoria unificante sintassi e semantica, la sua posizione a riguardo fu all'inizio molto ambigua. Nel lavoro "Syntactic Structures" Chomsky in più passaggi mostra la sua ambivalenza sulla questione.

> "The general problem of analyzing the process of *understanding* is thus reduced, in a sense, to the problem of explaining how kernel sentences are understood, these being considered the basic 'content elements' from which the usual, more complex sentences of real life are formed by transformational development." (p.92).

> "In other words, we should like the syntactic framework of the language that is isolated and exhibited by the grammar to be able to support semantic description, and we shall naturally rate more highly a theory of formal structure that leads to grammars that meet this requirement more fully." (p.102)

> "Grammar is best formulated as a self-contained study independent of semantics. In particular, the notion of grammaticalness cannot be identified with meaningfulness." (p.106)

Chomsky dunque supporta l'approccio preso dalle grammatiche trasformazionali nella misura in cui esse permettono di evidenziare differenze a livello *trasformazionale*[1] che sono invece celate a livello superficiale. Tuttavia col tempo la questione più importante divenne quella di trovare un fondamento comune fra sintassi e semantica.

Jerrold Katz e Jerry Fodor furono i primi a proporre di aggiungere una componente semantica alle grammatiche generative. Nella definizione di ciò che loro chiamarono *Projection Problem*, Katz e Fodor sottolinearono l'importanza di indagare in che modo il significato di un'espressione venga composta a partire dalle sue componenti più piccole. L'idea di base dell'approccio fu quella di definire il significato di una frase a partire dai passi eseguiti per la sua analisi sintattica. Tuttavia gli strumenti semantici utilizzati all'epoca della pubblicazione erano abbastanza semplici e la loro analisi non andò oltre la decomposizione di predicati unari. Ciononostante è importante sottolineare come questo fu un primo passo verso ciò che sarà poi formalizzato da Montague.

[1] Successivamente tali differenze saranno rielaborate all'interno di Deep Structure. Strutture queste ultime definite con lo scopo di unificare differenti strutture al suo interno.

Ovviamente un tale approccio non fu esente da critiche. Fra le molte critiche mosse, Lewis sottolineò come l'approccio di Katz e Fodor non era altro che un "sostituto" della *vera semantica* poiché esso si fonda sulla tacita assunzione che alcune competenze, ad esempio la possibilità di assegnare un giudizio di verità, siano innate in chi usa il linguaggio. Ma come linguisti, Katz e Fodor non erano interessati a tali aspetti e di conseguenza li avevano semplicemente assunti come universali ed innati. Occorsero anni comunque prima che i linguisti realizzassero l'importanza delle condizioni di verità e dei criteri di adeguatezza dell'analisi semantica.

2.2 La questione semantica dalla prospettiva logico-filosofica

Mentre per i linguisti l'analisi semantica si riduce ad un processo di traduzione, per i filosofi del linguaggio essa si realizza con sistemi formali definiti da assiomi e interpretati con la teoria dei modelli. I lavori sulla semantica in grammatica generativa si concentrarono maggiormente sulle procedure di traduzione fra le strutture sintattiche e semantiche, mettendo in secondo piano, o meglio tralasciando, la questione riguardante la definizione stessa di semantica. In filosofia del linguaggio invece la questione relativa alla "struttura formale"(logical form), necessaria per rappresentare aspetti del linguaggio, viene interpretata attraverso la definizione di schemi d'inferenza.

Fra i primi studiosi dei legami che coinvolgono la nozione di significazione nel linguaggio è doveroso citare Leibniz. La sua volontà di indagare le relazioni che sussistono fra linguaggio e realtà, linguaggio e pensiero, o fra linguaggio e conoscenza gli permisero di sviluppare un sistema formale di deduzione basato sul linguaggio *characteristica universalis* da lui stesso concepito. L'obiettivo di Leibniz era quello di introdurre una forma primaria di linguaggio simbolico che avesse una forma semplice per concetti semplici, e una forma non ambigua per strutture logiche di espressioni più complesse. Sfortunatamente Leibniz non riuscì a portare a termine il suo progetto, e occorsero due secoli prima di poter incontrare un'altra figura fondamentale nello sviluppo di questa disciplina.

Un cruciale avanzamento nella semantica composizionale è sicuramente riconducibile all'idea proposta da Gottlob Frege secondo la quale alcune espressioni possono essere interpretate come funzioni che ricevono come argomenti le denotazioni di altre espressioni. Per descrivere quest'approccio si consideri ad esempio la frase "Alice is happy". Qui il predicato *happy* può essere definito come una funzione caratteristica sull'insieme delle entità *happy* che applicata all'individuo *Alice* ne denota la sua estensione. Frege quindi fu il primo ad introdurre la distinzione fra senso e riferimento, o fra intensione ed estensione, concetto questo che vide poi diversi tentativi di formalizzazioni da parte dei filosofi del linguaggio.

La semantica formale, riconducile a Frege e al suo principio di compisizionalità, può essere definita come:

Definizione 2.2.1 *Il significato di un'espressione complessa è funzione del significato delle sue parti e del modo in cui esse sono sintatticamente combinate.*

Tuttavia il sistema proposto da Frege non era privo di difetti. Russell poco più tardi evidenziò[2] come la definizione di funzione proposta da Frege fosse poco robusta e sotto certe condizioni portasse ad una condizione paradossale. Russell propose allora un sistema di tipi che usò per imporre restrizioni sugli argomenti delle funzioni così da definire formalmente espressioni ben formate. In questo contesto si può sottolineare come Russell rivolse la sua attenzione più sui concreti riferimenti dei termini invece che al senso degli argomenti come diretto costituente delle proposizioni. Proseguendo su questa linea i riferimenti divennero presto centrali nella definizione formale della semantica.

In qualche modo il linguaggio deve sempre riferirsi al mondo. Riuscire a caratterizzare quest'aspetto è determinante per la comprensione di un'espressione linguistica, e la teoria dei modelli offre un'elegante soluzione. Tale approccio ricorre ad un'astrazione matematica, denominata modello, all'interno della quale ricondurre i significati dei riferimenti di un'espressione. Un modello rappresenta quindi il mondo, o per essere più precisi, una certa situazione o stato all'interno del quale i predicati corrispondono a funzioni che associano ad un'entità un valore di verità. In questo contesto, il valore semantico di un'espressione è riferito al valore di una proposizione che è vera o falsa in base alla situazione del mondo descritto dal modello. Non solo le espressione composte, ma anche altre componenti linguistiche più piccole, come nomi propri o verbi etc., ricevono lo stesso trattamento. Ma, se per i primi, i riferimenti possono essere consistentemente ricondotti a singole entità nel mondo, per i secondi, e specialmente per i verbi intransitivi, è richiesta una piccola estensione del trattamento.

Per modellare i verbi intransitivi come una definizione di proprietà, si associa il relativo valore semantico all'insieme di entità che possiedono quella determinata proprietà in un particolare mondo. Ora se da un lato questo modo di procedere permette di verificare la veridicità di un enunciato, dall'altro non discrimina fra i diversi significati che le espressioni linguistiche possono esprimere.

Una semantica estensionale come quella proposta da Tarski appiattisce il significato (intensione) sul riferimento (estensione), con la conseguente perdita della capacità di spiegare il diverso contenuto informativo o conoscitivo di espressioni linguistiche similmente estensionali. Soprattutto una semantica estensionale non è in grado di spiegare il valore di verità (estensione) degli enunciati che esprimono contesti intensionali, come gli enunciati modali ed epistemici. Tali enunciati, infatti, non sono composizionali rispetto all'estensione perché il loro valore di verità dipende dall'intensione e non dall'estensione di

[2] L'intento di Russell era in realtà contraddire la teoria degli insieme di Cantor, ma trovò utile contestualizzare la sua argomentazione in termini della definizione di funzione proposta da Frege.

alcune loro espressioni costituenti. Il trattamento formale di questi aspetti sono stati l'oggetto della ricerca condotta da Montague.

2.3 Montague

Benché la nozione di *mondi possibili* può essere ricondotta tranquillamente a Leibniz, occorre arrivare ai lavori di Tarski, Carnap, Kanger e Montague per poter avere una definizione formale della semantica. Soprattutto Tarski, con lo sviluppo della sua teoria dei modelli, permise i maggiori avanzamenti nella definizione dei linguaggi formali ancorando saldamente la definizione semantica con il concetto di *verità*. In seguito Carnap, pur apprezzando il lavoro di Tarski, vide nell'uso di un metalinguaggio estensionale una limitazione. Egli sviluppò allora un approccio alternativo dove la nozione di significato è equivalente alle condizioni di verità e dove i mondi possibili assumono la forma di descrittori di stati. Tali descrittori di stati sono insiemi contenenti espressioni atomiche, o la loro negazione, che hanno come obiettivo la descrizione dei mondi possibili di Leibniz. Analizzando la similarità dell'intensione come similarità di estensione in tutti i descrittori di stato, Carnap provò a fornire una semantica per la logica modale[3].

La ricerca su modalità e mondi possibili fu proseguita poi da Kanger e Kripke che distinsero i modelli dai mondi possibili. Quest'ultimi furono infine inglobati all'interno dei modelli così da poter collegare differenti assiomatizzazioni della logica modale a differenti relazioni di accessibilità fra i diversi mondi. Tuttavia fu solo con Montague e la sua logica intensionale che i lavori di Carnap e di Kripke e Kanger poterono essere unificati. Soprattutto i contributi di Montague sulla logica intensionale furono fondamentali per lo sviluppo della semantica formale per il linguaggio naturale.

La logica intensionale nasce quindi come sistema teorico all'interno del quale condurre un analisi filosofica. Montague infatti era riluttante all'idea che un linguaggio come quello inglese non potesse essere formale. Per lui non vi erano importanti differenze, almeno sul piano teorico, fra linguaggi naturali e linguaggi formali. Montague credeva fermamente nella possibilità di definire un sistema compatibile con le allora esistenti definizioni di linguaggi formali e le metodologie adottate per rappresentare i linguaggi naturali.

Punto fondamentale per la teoria di Montague fu il principio di composizionalità di Frege. Nella visione di Montague la sintassi è un algebra di "forme", mentre la semantica è un algebra costruita sui "significati". Nessun

[3] Carnap lega la semantica degli operatori modali di necessità e possibilità alle intenzioni delle espressione alle quali tali operatori sono applicati. Quest'approccio tuttavia è circoscritto alle modalità della logica pura, e non risulta applicabile oltre. Forse per queste limitazioni più tardi Carnap sposta i suoi interessi verso la pragmatica, affermando che esistono aspetti importanti del linguaggio che non possono essere espressi dalla sintassi e dalla semantica di una logica pura.

vincolo è imposto sulla natura di forma o significato, mentre composizionalmente vincolata è la relazione fra semantica e sintassi all'interno della quale deve esistere un omomorfismo fra gli elementi della sintassi e gli elementi della semantica. Questa generalità permette al sistema proposto da Montague di essere applicato indipendentemente dalla nozione usata per modellare il significato.

In *English as Formal Language*, per esempio, Montague prova a equipaggiare direttamente la lingua inglese con la semantica basata sulla teoria dei modelli; mentre nei due lavori successivi *Universal Grammar* e *The Proper Treatment of Quantification in Ordinary English*(PTQ) interpone nella traduzione un linguaggio formale per la logica intensionale e solo a quest'ultimo associa una semantica definita sui modelli. In PTQ, soprattutto, sia le espressioni che le loro interpretazioni ricevono una definizione ricorsiva. Ciò si discosti sia dalla semantica generativa sia dalla semantica secondo la tradizione interpretativa che invece costruiscono il significato di un'espressione a partire da uno o più livelli di descrizioni sintattiche da interpretare.

Altro elemento distintivo dell'approccio di Montague è la separazione fra la quantificazione e l'assegnazione delle variabili[4] che rende la sua grammatica diversa dalla logica del primo ordine e dalla quantificazione di Frege-Tarski[5]. Il cambio concettuale qui è determinante: la semantica è distinta dalla conoscenza della semantica. La semantica riguarda da un lato le condizioni di verità, e non la verità in sé, e dall'altro le relazioni che esse implicano senza considerare le sue interne rappresentazioni. In questo contesto la logica intesionale di Montague ha come obiettivo la costruzione di un sistema logico adeguato al trattamento di espressioni intensionali. Espressioni come "Barack belives that Ilary defeats Donald", per esempio, ricevono un'analisi come:

$$belive(barack, defeat(ilary, donald))$$

Tuttavia è possibile per un'altra espressione come "Barack does not belive that the president of the US defeats Donald" supporti le stesse credenze della precedente espressione. Supponiamo infatti che se Ilary fosse presidente degli Stati Uniti, allora Barack ne sia all'oscuro. Questa eventualità forza i riferimenti di *president of the US* e Ilary a coincidere sul simbolo logico "*ilary*", ma ciò implicherebbe che Barack creda un fatto e rinneghi l'altro, cioè una contraddizione. La soluzione a questo problema è considerare le credenze di Barack in termini di senso della clausola subordinata e non come un riferimento; da qui l'impiego di una semantica dei modelli basata sui mondi possibili ovvero l'estensione di un'espressione varierà in accordo con il mondo considerato.

[4] L'assegnamento è effettuato dall'operatore λ.

[5] E' importante sottolineare l'uso che Montague fa del processo di astrazione λ nella semantica della costruzione degli assegnameni sulle variabili. Montague fu innovatore nell'uso *attivo* dell'operatore λ per assegnare le variabili. Nel sistema proposto da Frege l'assegnamento era effettuato direttamente dai quantificatori, mentre nella proposta di Montegue l'assegnamento, effettuato tramite l'operatore λ, è trattarlo come entità a sé, ed è messo in relazione con la quantificazione.

Le intensioni sono definite come l'insieme di estensioni che un'espressione espressa all'interno dell'intensione ha in ognuno dei possibili mondi. Come avviene con tutti i verbi, che prendono clausole come loro argomenti, la relazione descritta è fra un individuo e l'intensione di una proposizione. Quest'ultima equivale all'insieme degli assegnamenti dei valori di verità definiti dalla valutazione della proposizione in ogni possibile mondo. Quindi le intensioni delle clausole rappresentano l'oggetto delle credenze di Barack, che non saranno più contraddittorie ma al massimo inaccurate rispetto al mondo reale. Ma ciò comporta che ogni proposizione necessariamente vera sarà vera in tutti i mondi, rendendo di fatto identiche le intensioni. Sembra dunque improbabile che la nozione di intensione definita in questo modo sia adeguata a catturare il senso di un espressione inteso come qualcosa differente al riferimento.

2.4 Semantica lessicale e pragmatica

Finora il significato delle espressioni è stato espresso in termini di riferimento. Una criticata abbastanza frequente all'approccio definito da Montague, ma più in generale alle semantiche formali, riguarda il problema dell'equivalenza fra le proposizioni. Quando il significato di una frase è espresso attraverso condizioni di verità, ogni sostituzione di elementi che preservi la verità in tutti i contesti intensionali ed estensionali è permessa. Ciò può portare alla formazione di strane e contro-intuitive proposizioni in linguaggio naturale. Come ad esempio nelle espressioni:

1. Should we take the lion back to the zoo?
2. Should we take the bus back to the zoo?

In queste espressioni una conoscenza extra-linguistica, spesso definita in termini di aspetti pragmatici del linguaggio, potrebbe suggerire di assegnare un valore "poco probabile" alla prima delle due espressioni e un valore certo alla seconda. E' importante notare qui come le difficoltà non risiedano nella conoscenza della lingua o delle strutture sintattiche, bensì nella possibilità di costruire il "corretto" schema concettuale rilevante per la sua interpretazione. Da ciò deriva la necessità di distinguere fra le rappresentazioni formali del significato dell'espressione, da un lato, e quelle degli aspetti pragmatici legati alla sua interpretazione dall'altro.

Rappresentare la componente pragmatica utile all'interpretazione equivale a definire i diversi livelli di conoscenza del mondo: lessicale, culturale/situazionale etc., ed soprattutto i loro legami. Questi livelli di conoscenza sono alla base delle diverse possibili costruzioni degli stati mentali. Tralasciando la visione proposta da Saussure sull'arbitrarietà del legame fra parola(morfema) e significato, è possibile associare a tale legame una connotazione pragmatica così da potergli associare un'interpretazione. Rimane tuttavia la questione sui confini interni della conoscenza, o meglio fra conoscenza lessicale ed enciclopedica.

Definire quale sia il limite fra conoscenza lessicale ed enciclopedica è difficile. Se non esistesse nessuna differenza o se essa fosse soltanto illusoria, come suggerito da Lakoff, allora tutta l'informazione verrebbe collassata all'intero del lessico con notevoli difficoltà nel trattamento composizionale del significato. L'alternativa sarebbe trattare queste conoscenze come due entità distinte e utilizzare due meccanismi separati per la loro manipolazione. Detto in altre parole impiegare una metodologia per gli aspetti composizionali, e una per l'interpretazione concettuale.

Nelle sezioni precedenti, infatti, l'attenzione per la composizione del significato è stata rivolta principalmente sulla definizione della sua dinamica interna al sistema formale in uso. Guidati dal principio di composizionalità. gli approcci descritti costruivano il significato di un'espressione a partire dalle sue componenti più piccole. Tuttavia in una successiva definizione proposta da Fodor e Pylyshyn si sottolinea un'ulteriore importante aspetto di questo principio:

"A lexical item must make approximately the same semantic contribution to each expression in which it occurs."

La questione è qui spostata sul contributo della singola componente linguistica, aldilà del suo riferimento e del suo valore di verità, alla costruzione del significato dell'espressione. Lo studio del significato delle più piccole parti costituenti le espressioni linguistiche prende il nome di semantica lessicale.

Il significato delle unità lessicali[6] è assunto come empiricamente giustificabile attraverso il significato di espressione di cui fanno parte. In questa assunzione l'approccio composizionale diventa quindi un utile strumento di indagine. Soprattutto, l'uso di tale principio permette di non assimilare l'analisi lessicale e la relativa grammatica a qualche forma generica di enumerazione fra le possibili relazioni di entità indipendenti. Ipotesi questa dell'indipendenza che difficilmente trova riscontro non solo negli studi psicologici-cognitivi ma anche in quelli di natura prettamente linguistica. Anzi assumere un certo grado di dipendenza potrebbe facilitare l'analisi delle anomalie e delle contraddizioni.

All'interno della semantica lessicale infatti dovrebbero trovare posto gli elementi per rivelare eventuali contraddizioni. Infatti se nei sistemi formali l'incompatibilità fra due costituenti è trattata essenzialmente attraverso un sistema di tipi che impedisce la costruzione di strutture logiche paradossali. Estendere tale approccio anche sul piano pragmatico richiede una conoscenza che vada oltre la semplice organizzazione gerarchica, e che descriva la realtà con conoscenza extra-linguistica espressa in termini di assiomi.

Si consideri la combinazione nome-aggettivo come proposta da Quine per gli esempi "*red apple*" e "*pink grapefruit*". Nel primo caso l'aggettivo *red* si riferisce all'esterno del frutto mentre nel secondo caso *pink* al suo interno. Il contributo offerto dall'aggettivo non è facilmente semplificabile in una forma

[6] Rimane comunque aleatoria la definizione formale di cosa sia un'unità lessicale e soprattutto quale sia il suo contenuto.

prefissata in quanto esso varia in base alla sua composizione. Alcuni autori come Montague, Keenan, e Kamp proposero di considerare gli aggettivi come funzioni ad-nominali; cioè come elementi che mutano le proprietà del nome al quale sono associati in un particolare modo. Ma ancora un tale approccio conduce alla necessità di dover specificare tutti i possibili casi e si perde quindi la possibilità di risolvere la questione della sistematicità della capacità linguistica. Più recentemente Pustejovsky propose nella sua teoria di lessico generativo[7] di scomporre il significato lessicale in un insieme di dimensioni semantiche, ovvero unità atomiche di significato e concettualizzazione.

Per Pustejovsky, il semplice enumerare i sensi non permette di catturare le (presunte) relazioni semantiche fra i differenti sensi di un espressione polisemica. Egli infatti sottolinea come ad essere modificate sia in realtà la struttura semantica, denominata *struttura di qualia*, ad essa associata. Questa struttura, articolata su più dimensioni, definisce una certa entità come un insieme di fatti: le dimensioni posso essere relative alle parti costituenti, aspetti funzionali, di contesto, o riguardanti le origine del termine. Tuttavia, anche se tale approccio sembra funzionare bene per alcuni esempi, esso sottende alcuni problematiche importanti.

I problemi maggiori nell'applicare l'analisi proposta da Pustejovsky riguardano il processo d'inferenza e il piano ontologico. Tale teoria attribuisce infatti una relazione inferenziale fra le parti in esame. Nella frase "*the red apple*", per esempio, l'inferenza porta a attribuire la caratteristica di rosso alla buccia della mela. Essenzialmente questa metodologia può essere assimilata ad un processo di "unificazione" monotono operato su informazione non confutabile. Le inferenze quindi si trasformano in implicazioni necessarie, ipotesi questa poco plausibile nella realtà. Ma quanto appena discusso non è l'unico problema evidenziato nella teoria. Ulteriore fonte di preoccupazione, forse molto più difficile da trattare, riguarda invece l'individuazione degli aspetti salienti da utilizzare nelle strutture di qualia. Nell'esempio precedente si evidenziano alcune caratteristiche di un aggettivo, ma ciò che può essere rilevante per un aggettivo che indica un colore può non essere rilevante per un altro genere di aggettivo che descrive magari il gusto. Risulta in generale molto difficile selezionare sia le dimensioni sulle quali costruire le strutture semantiche sia selezionare alcuni loro sotto-insiemi per operare modifiche al loro interno.

Nonostante le intuizioni alla base dell'approccio sviluppato da Pustejovsky meriterebbero ulteriori approfondimenti, le problematiche appena descritte hanno limitato non poco la diffusione di questa teoria.

[7] La teoria di Pustejovsky può essere vista come una variante dell'approccio basato sulla generazione selettiva. Tale approccio descrive i vincoli semantici che una certa parola impone sull'ambiente sintattico. Per esempio il verbo *to eat* preferisce come suo oggetto diretto un cibo. Quindi nella frase *John eats the orange* la scelta del giusto senso per *orange* dovrebbe preferire la sua accezione di frutto invece che quella di colore.

2.5 Semantica del discorso

La semantica proposta da Montague non è capace di costruire forme logiche per espressioni linguistiche che si compongono di più frasi. All'interno del sistema definito da Montague, ad esempio, non è possibile definire il giusto riferimento anaforico fra gli elementi contenuti in una successione di frasi. Si consideri la seguente espressione:

A guy walked in. He bought a book.

con le associate forme logiche

1. $\exists x(guy(x) \wedge walk - in(x)) \wedge \exists y(book(y) \wedge buy(z, y))$
2. $\exists x(guy(x) \wedge walk - in(x) \wedge \exists y(book(y) \wedge buy(x, y)))$

La semplice combinazione delle forme logiche come in (1) porterebbe ad un'incorretta predizione permettendo ad un ragazzo qualsiasi di comperare un libro essendo la variabile z non quantificata. La seconda forma logica gestisce invece il caso precedente ma la sua costruzione richiederebbe l'estensione dello scopo della quantificazione oltre i limiti della frase; ciò implicherebbe un notevole aumento della complessità dell'analisi. Il punto cardine della questione è qui la definizione del contesto del discorso. La prima espressione definisce infatti il contesto all'interno del quale la seconda espressione deve essere interpretata. Lo scopo principale dell'approccio denominato *semantica dinamica* è quello di fornire un adeguato trattamento alle informazioni di contesto.

La semantica dinamica ha come obiettivo la descrizione delle relazioni fra contesti, o come definito da Kamp, del *context change potential*. L'interpretazione è descritta in termini di relazione fra contesti iniziali ed finali. Qui il contesto iniziale può essere immaginato come l'insieme di funzioni d'assegnamento che rendono vero il contenuto del discorso all'interno di un prefissato modello. Data allora una nuova espressione E, il contesto finale è il sottoinsieme di tali funzioni che rendono E vera nel modello. In questo modo è possibile trattare elegantemente la situazioni mostrata dall'esempio precedente.

Attraverso l'utilizzo di *riferimenti del discorso*, è possibile definire un principio di accessibilità che può essere facilmente interpretato come presenza di un certo assegnamento per la variabile. In altre parole, nell'esempio precedente è possibile garantire attraverso l'assegnamento alla variabile x che il ragazzo che entra sia effettivamente colui che compra il libro.

Nella teoria della rappresentazione del discorso (DRT) Kamp e Relye formalizzarono le loro intuizioni sulla semantica dinamica. Originariamente la teoria fu sviluppata da Kamp per trattare con la risoluzione dei pronomi e le relazioni temporali, successivamente Kamp e Relye definirono la versione standard incorporando una semantica degli eventi di Davidson. DRT usa un rappresentazione intermedia chiamata struttura del discorso(DRS) per esplicitare aspetti del linguaggio che sono sensibili al contesto: anafora, proposizioni ellittiche e presupposizione. La sintassi delle DRS e dei vincoli che possono essere definiti al loro interno è definita ricorsivamente su un insieme di variabili (del primo ordine) e simboli di predicati insieme con la loro arietà.

Definizione 2.5.1 *Le condizioni su una DRS sono definite come*

1. *dato un simbolo di predicato n-ario p, e un insieme di riferimenti* $\{r_1, \ldots, r_n\}$, *allora* $p(r_1, \ldots, r_n)$ *è una condizione;*
2. *data una coppia di riferimenti* r_1 *e* r_2 *allora* $r_1 = r_2$ *è una condizione;*
3. *data una DRS s, allora* $\neg s$, $\Box s$, *e* $\Diamond s$ *sono condizioni;*
4. *data una coppia di DRS s e s', allora* $s \vee s'$, $s \implies s'$ *sono condizioni;*
5. *dato un riferimento r e una DRS s, allora* $r : s$ *è una condizione;*

Le condizioni possono essere semplici o complesse. Nel primo caso esse si riferiscono a proprietà dei riferimenti del discorso o a qualche loro forma di relazione. Nel secondo caso invece esse sono caratterizzate dall'introduzione di DRS subordinate. L'ultima condizione infine può essere considerata come un operatore modale, che assegna al riferimento r il mondo rappresentato dalla DRS s. Quest'ultima condizione può quindi essere definita formalmente come:

Definizione 2.5.2 *Dati un insieme di riferimenti R e un insieme di condizioni* Γ, *una DRS s è definita come una coppia*

$$s = \langle R', \Gamma' \rangle$$

per qualche $R' \subseteq R$ *e* $\Gamma' \subseteq \Gamma$

Le DRS sono quindi definite come composte da due parti: da un lato un insieme di riferimenti del discorso, e dall'altro un insieme di condizioni che vincolano l'interpretazione dei riferimenti. Secondo questa definizione la DRS per l'espressione "John though that he does not give Anna a book" è la seguente:

$$\begin{aligned} s_0 =& \langle \{z, e'\}, \{book(z), give(e', x, y, z)\} \rangle \\ s_1 =& \langle \emptyset, \{\neg s_0\} \rangle \\ s_2 =& \langle \{x, y, e, c\}, \{john(x), anna(y), think(e, x, c), c : s_1\} \rangle \end{aligned}$$

Qui è importante notare come i nomi propri introducono riferimenti del discorso a livello globale, mentre le frasi nominali introducono riferimenti solo localmente. I pronomi poi non introducono nuovi riferimenti, ma invece si legano a riferimenti già presenti.

L'interpretazione delle DRS si basa in modo indiretto sulla teoria dei modelli. Attraverso una funzione di traduzione, le DRS sono trasformate in formule del primo ordine che condividono con le DRS lo stesso vocabolario di predicati e variabili.

Definizione 2.5.3 *Sia* $w \in W$ *una variabile indicante un possibile mondo, allora la funzione f è definita come*

- $f(w, \langle \{x_1 \ldots x_n\}, \{\gamma_1 \ldots \gamma_n\} \rangle) \doteq \exists x_1 \ldots \exists x_n (f(w, \gamma_1) \wedge \cdots \wedge f(w, \gamma_n))$;
- $f(w, p(x_1, \ldots, x_n)) \doteq p(w, x_1, \ldots, x_n))$;
- $f(w, x_1 = x_2) \doteq x_1 = x_2$;

- $f(w, \neg s) \doteq \neg f(w, s)$;
- $f(w, s_1 \vee s_2) \doteq f(w, s_1) \vee f(w, s_2)$;
- $f(w, \langle \{x_1 \dots x_n\}, \{\gamma_1 \dots \gamma_n\} \rangle \implies s) \doteq \forall x_1 \dots \forall x_n((f(w, \gamma_1) \wedge \dots \wedge f(w, \gamma_n)) \rightarrow f(w, s))$;
- $f(w, \Diamond s) \doteq \exists v(p(w, v) \wedge f(v, s))$
- $f(w, \Box s) \doteq \forall v(p(w, v) \rightarrow f(v, s))$
- $f(w, v : s) \doteq (p(w, v) \wedge f(v, s))$

Questa trasformazione è stata proposta da Bos, ed è basata sul lavoro di Kamp and Reyle ma estesa con la proposta di Moore per il trattamento degli operatori modali. Qui ogni riferimento del discorso diviene un quantificatore, mentre le condizioni diventano invece una congiunzione di formule del primo ordine. Le implicazioni in DRS diventano implicazione materiali nella logica del primo ordine. Si noti quindi come tale trasformazione risulti lineare nella lunghezza dell'input. Infine Kamp e Reyle integrarono nella loro proposta la DRT con la semantica degli eventi proposta da Davidson; in questo modo i riferimenti del discorso poterono rappresentare eventi ai quali era possibile riferirsi tramite espressioni anaforiche o relazioni temporali.

Recentemente Lascarides e Asher hanno suggerito di estendere il linguaggio della DRT con relazione retoriche per poter meglio modellare le intenzioni di chi parla. Queste relazioni possono essere immaginate come un insieme di informazioni accessorie utili nell'azione della comunicazione per poter ricostruire la coerenza del discorso. Detto in altri termini, una frase come "*John took the train to Rome. He works there.*" è perfettamente accettabile, mentre "*John took the train to Rome. He plays guitar.*" è comunemente identificata come discorso poco coerente. Nella teoria di rappresentazione del discorso segmentato(SDRT) Lascarides e Asher descrivono la coerenza del discorso attraverso una struttura gerarchicamente costruita attraverso relazioni retoriche.

2.6 Note sulla teoria di rappresentazione del discorso

Nella sua formulazione originaria la teoria di rappresentazione del discorso fu oggetto di molte critiche. Infatti se accostata ai formalismi logici presenti in letteratura si sarebbe portati a credere che tale approccio non sia nulla di nuovo, e che anzi sia per sua natura limitato se paragonato ad essi. Soprattutto i limiti evidenziati nel trattare alcuni aspetti del linguaggio, al di fuori della risoluzione dell'anafora, richiesero successive estensione della teoria originaria. Lo scopo di queste note è quello di chiarire, seguendo alcuni lavori di Hess, come alcuni punti della DRT non siano di così immediata traduzione in logica e che anzi tali punti richiedono uno spostamento del ragionamento su un meta livello per poter essere correttamente trattati.

L'analisi del discorso, descritta nella sezione precedente, si conclude con la produzione di una DRS assimilabile alla rappresentazione di un modello[8]. In questo modello il ruolo di una semplice DRS è la descrizione di proprietà e relazioni fra i riferimenti del discorso, mentre quello delle DRS più complesse è spesso assimilato a regole d'inferenza. Queste regole potendo descrivere proprietà di entità *potenziali* non dovrebbero essere intese con l'accezione di funzioni di espressioni universalmente quantificate proprie della logica. Invece esse dovrebbero essere trattate come *istruzioni* da applicare su potenziali oggetti che potrebbero apparire in futuro nel modello. Oggetti, si noti bene, ai quali è possibile accedere esclusivamente attraverso i riferimenti del discorso. Da qui le prime critiche alla DRT furono proprio sui riferimenti del discorso e sul ruolo da loro svolto.

L'utilizzo dei riferimenti del discorso non propone nulla di nuovo. Tali riferimenti si comporterebbero allo stesso modo della quantificazione esistenziale in logica. Nella DRT infatti essi possono essere rappresentativi di qualche entità del mondo reale, possono essere co-referenziali, ma non possono riferirsi a più di un oggetto esterno. Nell'analizzare una semplice espressione come *"a boy read a red book. He liked it."*, DRT crea un nuovo riferimento del discorso per ogni frase nominale indefinita, questi riferimenti verranno poi possibilmente co-referenziati con qualche potenziale riferimento. Una DRS per l'espressione precedente può essere definita come:

$$\langle \{r_1, r_2, r_3, r_4\}, \{boy(r_1), red(r_2), book(r_2), \\ read(r_1, r_2), r_3 = r_1, r_4 = r_2, like(r_3, r_4)\}\rangle$$

Una volta risolti tutti i riferimenti e specificate tutte le relazioni è possibile usare la DRS all'interno di una base di conoscenza per recuperare i dati d'interesse e procedere con l'inferenza. Tuttavia è prassi comune tradurre le DRS in qualche linguaggio formale eliminando dove possibile le informazioni specifiche per il solo discorso.

Scegliendo un opportuno linguaggio si evidenzia l'esistenza di alcuni parallelismi fra i due linguaggi. Almeno ad una prima analisi, la diretta traduzione dei rispettivi elementi è immediata, e si sarebbe potarti a pensare che l'idea stessa dei riferimenti del discorso sia di per se ridondante all'interno del livello logico. Si consideri, ad esempio, come linguaggio formale *datalog*, che sarà introdotto nel capito 6, l'analisi per l'espressione precedente può essere tranquillamente definita come un insieme di *fatti* logici:

$$boy(r_1).$$
$$red(r_2).$$
$$book(r_2).$$
$$read(r_1, r_2).$$
$$like(r_1, r_2).$$

[8] Tale modello si riferisce essenzialmente alla situazione espressa dal discorso ma può essere comunque esteso oltre i limiti del discorso con informazioni non linguistiche provenienti da qualche sorgente esterna d'informazione.

La traduzione appena proposta prevede di definire un fatto per ogni semplice condizione nella DRS e per ogni loro riferimento una costante di *Skolem*. Soprattutto è importante notare come i riferimenti ausiliari,r_3 e r_4, siano spariti completamente e che la forma sintattica delle costanti di Skolem, il cui scopo si estende all'intero programma logico, garantisca la co-referenzialità. Hess allora sottolinea come la conclusione che si sarebbe portati a trarre è che il ruolo dei riferimenti del discorso nelle DRS di frasi multiple sia lo stesso delle constanti di Skolem in un programma logico costituito da clausole multiple: cioè attestare l'*esistenza*.

Tuttavia, i riferimenti possono a volte richiedere una traduzione su variabili. Si consideri ad esempio le espressioni:

If a man owns a donkey he beats it.
If a man owns a donkey he is happy.

con le associate DRS:

- $\langle\{\}, \{\langle\{r_1, r_2\}, \{man(r_1), donkey(r_2), own(r_1, r_2)\}\rangle \rightarrow \langle\{r_3, r_4\}, \{r_3 = r_1, r_4 = r_2, beat(r_3, r_4)\}\rangle\}\rangle$
- $\langle\{\}, \{\langle\{r_1, r_2\}, \{man(r_1), donkey(r_2), own(r_1, r_2)\}\rangle \rightarrow \langle\{r_3\}, \{r_3 = r_1, happy(r_3)\}\rangle\}\rangle$

le cui traduzioni in datalog pur portano comunque alle stesse condizioni di verità richiedono un trattamento diverso per i riferimenti:

$$beat(X, Y) \coloneq man(X), donkey(Y), own(X, Y).$$
$$happy(X) \coloneq man(X), donkey(Y), own(X, Y).$$

A differenza dell'esempio precedente, qui, la condizione complessa di implicazione fra DRS è tradotta in una regola logica. I riferimenti al suo interno sono espressi con variabili, la cui interpretazione è universalmente quantificate all'interno delle regole.

Dagli esempi appena discussi, si potrebbe quindi assumere che i riferimenti del discorso di espressioni non generiche e generali si comportino come costanti, mentre quelli le cui DRS si riferiscono a frasi generiche debbano tradursi con variabili. Sebbene quest'ultimo punto permetta di gestire correttamente l'accessibilità intra-clausola dei riferimenti, rimane preclusa la possibilità di trattare la loro accessibilità inter-clausola. Tuttavia, come discusso precedentemente, DRT fu originariamente progettato con l'intento di trattare proprio tali fenomeni, evidenziando come l'analisi intra e inter-clausola richieda un approccio differenziato. Nell'analisi intra-clausola infatti i riferimenti espressi con variabili sono sufficienti a trattare correttamente la co-referenzialità. Mentre trattare il secondo caso richiede regole aggiuntive per poter rappresentare le dinamiche che in DRT determinano il passaggio di un riferimento da una DRS ad una più esterna. Proprio quest'ultimo aspetto è l'elemento distintivo della teoria.

Concludendo il modo più "corretto" per definire i riferimenti del discorso è in termini di *meta*-variabili riferite ad un sottostante linguaggio oggetto.

Queste meta-variabili, necessarie per garantire la co-referenzialità, possono rappresentare costanti, variabili, o funzioni, ma ciò che occorre evidenziare è come esse siano utilizzate all'interno delle regole che governano il passaggio dei riferimenti da una DRS ad una più esterna. Tali regole sono la parte essenziale della DRT e rendono i riferimenti di per se indispensabili. Si può quindi descrivere il modello del discorso prodotto da DRT come una meta-struttura, sopra qualche linguaggio oggetto per la rappresentazione della conoscenza, la cui natura non è propriamente estensionale bensì intensionale.

2.7 Davidson e la reificazione degli eventi

Finora la discussione sugli approcci alla semantica ha evidenziato esclusivamente i principi alla base di queste teorie senza valutare le loro reali applicazioni. L'uso indiscriminato di logiche di alto livello, come la logica modale, intensionale o dei mondi possibili è largamente evitato nella comunità della semantica computazionale per via della complessità di queste logiche. Anche l'uso concreto che viene fatto del linguaggio λ è generalmente limitato alla semplice costruzione, in modo composizionale e guidato dall'analisi sintattica, di formule del primo ordine.

Per evitare di ricorrere a logiche più espressive, la soluzione più utilizzata è reificare eventi e mondi all'interno della logica del primo ordine. Per esempio, l'espressione "John eats the sandwich quickly" potrebbe ricevere una forma logica del secondo ordine simile a:

$$quickly_{eating}(eat(John, sandwich))$$

Davidson fu il primo a suggerire una diversa rappresentazione in cui l'evento è reificato. Per lui infatti l'espressione può essere tranquillamente espressa nella seguente forma del primo ordine:

$$\exists e\ eat(e, john, sandwich) \wedge quickly_{eating}(e)$$

Anche espressioni che esprimono condizioni di stato possono essere facilmente reificate. Tuttavia esistono alcune situazioni come ad esempio quelle contenenti l'idea di eventualità che non trovano una *giusta* traduzione seguendo quest'approccio:

Possibly every bird flies

$$\forall x\ \exists e\ bird(x) \implies (fly(e, x) \wedge possibly(e))$$

Il corretto trattamento di quest'ultimo esempio probabilmente va oltre le intenzioni di Davidson. Tuttavia il suo approccio risulta molto utile anche nella rappresentazione logica delle diverse forme temporali che un verbo può assumere.

L'evoluzione di questo approccio, comunemente indicato con neo-Davidson, estese l'idea iniziale così da rappresentare ulteriori aspetti dell'espressione.

Parsons propose di far diventare gli argomenti dei predicati delle relazioni binarie fra variabili d'evento ed entità come

$$\exists e\ kick(e) \wedge agent(e, x) \wedge patient(e, y)$$

Ciononostante con il tempo si preferì mantenere un approccio più neutrale e flessibile verso l'annotazione semantica indicando con generici predicati $arg_1(e, x), \ldots, arg_N(e, z)$ gli argomenti dei verbi. Questo tipo di annotazione, sebbene meno informativo, risulta essere più flessibile in quanto capace di gestire la presenza di un numero variabile di argomenti.

2.8 Rappresentazione astratta del significato

Alla fine degli anni 90 Langkilde e Knight presentarono una semplice struttura basata su alberi per rappresentare la semantica di un'espressione naturale. Denominata *rappresentazione astratta del significato*(AMR), la teoria si discosta da altri formalismi, come ad esempio DRT, per un diverso trattamento della negazione, la mancanza di un adeguato trattamento della quantificazione universale, e soprattutto per l'assenza di una semantica formale basata sulla teoria dei modelli. Tuttavia, come sottolinearono Blackburn e Bos, le finalità di una teoria per la semantica del linguaggio naturale possono essere circoscritte alla produzione di una rappresentazione sulla quale sia possibile fare inferenza su sistemi di deduzione automatica. Forse proprio per questo motivo AMR ha acquisito negli ultimi tempi una crescente attenzione: infatti AMR è sia facile da interpretare nei costrutti che genera, sia veloce da annotare su grandi collezioni di testi.

Le strutture di AMR sono semplici e possono essere espresse con grafi diretti aciclici, che, essendo caratterizzati da un unico nodo *root*, sono per loro natura degli alberi. Il seguente vocabolario è utilizzato nella definizione della sintassi:

- variabili x, y, etc,e costanti $c_1, \ldots, c_n$;
- eventi e concetti descritti da proprietà $P_1, \ldots, P_m$;
- ruoli $R_1, \ldots, R_q$;
- un'operazione di istanziamento "/" tale che x/P_i denota che x è un'istanza di P_i;
- un'operazione di assegnamento di ruolo ":".

Indicando poi con A_i le singole rappresentazioni, la sintassi di una AMR può essere definita come:

Definizione 2.8.1 $A ::= c|(x/P)|(x/P : R_1A_1 \cdots : R_nA_n)$

In questa definizione i differenti tipi possono essere tutti considerati come denotazioni di proposizioni differenti. Un'espressione come "*Mr. Obama gave a speech.*" può essere espressa in AMR come:

$$(e/give\ :ARG0(x/person\ :named\ \text{“}Mr\ Obama''\ ):ARG1(y/speech))$$

Se si restringe AMR al contesto di un'analisi senza polarità è possibile poi sfruttare un'altra interessante sua caratteristica: l'inversione dei ruoli. Tale procedura, $R(x,y)\hat{=}R_{of}(y,x)$, permette di esprimere costrutti AMR semanticamente equivalenti ma strutturalmente differenti.

Sulla sintassi appena proposta è possibile definire una funzione ricorsiva che trasforma i costrutti AMR in logica del primo ordine. Tuttavia, per ottenere un corretto trattamento degli assegnamenti di scopo, i ruoli non sono tradotti direttamente ma sono invece convertiti in espressioni λ. La semantica di base per l'AMR può essere definita come:

Definizione 2.8.2 •

- $||c,\phi|| = \phi(c)$
- $||(x/P),\phi|| = \exists x(P(x) \wedge \phi(x))$
- $||(x/P\ :R_1A_1\ldots\ :R_nA_n),\phi|| = \exists x(P(x) \wedge ||A_1,\lambda y.R_1(x,y)|| \wedge \cdots \wedge ||A_n,\lambda y.R_n(x,y)|| \wedge \phi(x))$

Qui ϕ denota un'espressione λ. La struttura che ne risulta è una formula chiusa poiché la traduzione impedisce la presenza di variabili libere. Tuttavia attraverso la relazione di polarità è possibile esprimere il concetto di negazione.

Finora AMR è stata trattata nella sua forma più semplice. La relazione di polarità è definita fra un concetto(che sarà negato) e la costante "−". La presenza di tale relazione richiede uno specifico trattamento nell'assegnamento di un'interpretazione che impone delle modifiche sia sulla sintassi sia sulla semantica.

Definizione 2.8.3 $A ::= c|(x/P)|(x/P:R_1A_1\cdots:R_nA_n)|$
$(x/P:R_1A_1\cdots:R_nA_n\ :polarity-)$

e

Definizione 2.8.4 •

- $||c,\phi|| = \phi(c)$
- $||(x/P),\phi|| = \exists x(P(x) \wedge \phi(x))$
- $||(x/P\ :R_1A_1\ldots\ :R_nA_n),\phi|| = \exists x(P(x) \wedge ||A_1,\lambda y.R_1(x,y)|| \wedge \ldots \wedge||A_n,\lambda y.R_n(x,y)|| \wedge \phi(x))$
- $||(x/P\ :R_1A_1\ldots\ :R_nA_n\ :polarity-),\phi|| = \neg\exists x(P(x)\wedge||A_1,\lambda y.R_1(x,y)||\wedge \ldots \wedge||A_n,\lambda y.R_n(x,y)|| \wedge \phi(x))$

Se la negazione è assegnata al concetto radice allora il suo scopo si estende su tutti i sottostanti quantificatori esistenziali. In altri casi essa potrebbe essere invece più interna, e la proposizione costruita equivarrebbe ad una situazione in cui una certa cosa *non* esiste.

Il trattamento della quantificazione universale invece è decisamente più problematico. Sebbene in generale i costrutti AMR non possono esprimere

quantificazione universale, è possibile, attraverso la tecnica di estrazione dei concetti e la relazione di polarità, rappresentare alcune espressioni universalmente quantificate. Per esempio, un espressione come "*every boy played*" può essere rappresentata come:

$$\neg\exists x(boy(x) \wedge \neg\exists e(play(e) \wedge ARG0(e, x)))$$

che è logicamente equivalente alla più tradizionale:

$$\forall x(boy(x) \implies \exists e(play(e) \wedge ARG0(e, x)))$$

Tuttavia, data la natura delle AMR, non più di un quantificatore universale è esprimibile in questo modo, da qui l'impossibilità di esprimere espressioni tipo *every boy played every guitar*.

Concludendo AMR è diversa dalle teorie formali di rappresentazione della semantica presentate in questo capito. Se da un lato essa permette di esplicitare alcuni aspetti della struttura semantica, dall'altro l'impossibilità di rappresentare scopi rende necessario un differente modo di trattare la negazione. Quest'ultima può portare a cambiamenti nel significato quando associato all'inversione dei ruoli.

2.9 Sommario e riferimenti per approfondimenti

Nella tradizione linguistica i primi lavori sulla semantica possono essere riferiti ai lavori di Bréal ed in particolare a [13]. Successivamente la discussione fu posta leggermente in secondo piano a cause della posizione abbastanza ambigua di Chomsky. I passaggi riportati sono estratti dal lavoro [26]. Solo dopo il lavoro di Kats e Fodor [77] la semantica riprese un ruolo di interesse all'interno della comunità linguistica. Nella definire il Projection Problem gli autori evidenziarono l'importanza di indagare maggiormente il legame che sussiste fra sintassi e semantica.

Parallelamente alla prospettiva linguistica la semantica è stata analizzata lungo una dimensione di carattere prettamente filosofico. L'analisi filosofica del problema della significazione ha una storia più antica della sua corrispettiva linguistica. Leibniz col suo *Calculus ratiocinator* iniziò una lunga tradizione che ebbe il suo culmine con l'avvento della logica matematica. In questo contesto molti dei principi fondanti sono stati delineati dal lungo lavoro di Frege. Soprattutto in [49] Frege distinse per la prima volta il senso dal riferimento. Le differenze fra intensione ed estensione furono oggetto di tentativi di formalizzazioni da parte dei filosofi del linguaggio come Russell e Tarski, solo per citarne alcuni. Tuttavia solo con il contributo di Montague si poté iniziare a modellare la questione semantica in un opportuno formalismo logico.

Fra i lavori alternativi alla teoria dei modelli definiti da Tarski vi fu l'approccio definito da Carnap in [20, 21]. Al suo interno il significato era sempre caratterizzato da un valore di verità ma la definizione di mondi possibili era

qui realizzata in termini di descrittori di configurazione di stati. Nei suoi lavori Carnap provò a fornire una semantica per la logica modale all'interno del contesto dei mondi possibili, questa ricerca fu poi proseguita da Kanger in [75, 74] e Kripke [79, 80] che distinsero i modelli dai mondi possibili che furono infine inglobati all'interno dei modelli. Tuttavia fu solo con Montague e la sua logica intensionale che i lavori di Carnap e Kripke poterono essere unificati.

Il lavoro di Montague sul trattamento formale della semantica del linguaggio naturale è concentrato in tre lavori *English as Formal Language* [110], *Universal Grammar* [111], ed infine *The Proper Treatment of Quantification in Ordinary English* [112].

Per la semantica lessicale approfondimenti sulla rivisitazione della composizionalità definita da Fodor e Pylyshyn può essere trovata in [48]. Il primo riferimento contenente una discussione sul contrasto fra i casi *red apple* e *pink grapefruit* può essere trovato nelle note di Quine in [128]. Approfondimenti sugli argomenti proposti da Pustejovsky contro l'enumerazione, che può essere vista come una variante dell'approccio della generazione selettiva, possono essere trovati in [125, 126, 127].

Il riferimento più importante per la reificazione degli eventi è sicuramente il saggio di Davidson [32], ma anche le note successive [31] sono un'interessante lettura. Mentre per approfondire la forma proposta da Parsons, che prende il nome di neo-Davidson, è possibile consultare [141].

La teoria di rappresentazione del discorso è stata originariamente sviluppata da Kamp in [72] per trattare la risoluzione dei pronomi e le relazioni temporali. Mentre la versione standard della teoria, incorporante anche la semantica degli eventi di Davidson, è invece proposta successivamente in collaborazione con Reyle in[73]. Il lavoro più recente di Lascarides e Asher su *Segmented discourse representation theory* è [83]. Infine le note riportate sul ruolo dei riferimenti del discorso sono un semplice accenno alla discussione proposta da Hess in [63].

La teoria di rappresentazione astratta del significato è stata dapprima introdotta con il lavoro di Langkilde e Knight [82]. Successivamente Bos proseguì lo sviluppo di questa teoria nei lavori [8, 11].

3

Modelli neurali per il linguaggio naturale

In netto contrasto con la trattazione formale della tradizione di Chomsky, l'approccio conosciuto come *word embeddings*, nel seguito denominato con il termine *vettori di parole*, si basa sull'assunzione che le informazioni contestuali, da sole, offrano un'adeguata rappresentazione delle entità linguistiche.

Lo scopo dell'analisi distribuzionale è l'acquisizione del significato sfruttando la distribuzione delle entità linguistiche all'interno di collezioni di testo. L'ipotesi fondamentale alla base di questo approccio è spesso parafrasato con l'espressione "parole simili nel significato tendono ad essere caratterizzate da contesti simili", o con "parole che occorrono nello stesso contesto tendono ad avere lo stesso significato". L'idea qui è di evidenziare una correlazione fra le caratteristiche della distribuzione associata ad una parola ed il suo significato, in modo da individuare quest'ultimo in funzione della prima.

Questo capitolo introduce i principi di base dell'analisi distribuzionale e discute i più recenti modelli per il linguaggio costruiti con l'ausilio delle reti neurali. Questi modelli, conosciuti appunto con il nome di modelli neurali, saranno analizzati nelle varie sezioni cercando dove possibile di evidenziarne pregi e difetti.

3.1 Ipotesi di distribuzione del significato

La semantica distribuzionale trova le sue origini teoriche nella linguistica strutturalistica e nella filosofia del linguaggio ed in particolare nei lavori di Harris, Firth, e Wittgenstein risalenti alla metà del secolo scorso. Infatti i primi tentativi di usare qualche genere di rappresentazione per quantificare la similarità semantiche possono essere ricondotti ai lavori sulla semantica differenziale di Charles Osgood negli anni 60 che furono successivamente estesi nel campo dell'intelligenza artificiale nei primi anni 80.

L'idea di Harris proponeva di raggruppare gli elementi delle classi linguistiche di base secondo i loro comportamenti distribuzionali. Se, ad esempio, due parole w_1 e w_2 tendono ad essere accompagnate da una terza parola w_3,

è ragionevole assumere che esse appartengano alla stessa classe linguistica. Da questa considerazione Harris propose di caratterizzare il linguaggio nella sua interezza senza la necessità di ricorrere ad aspetti esterni come definizioni ontologiche e funzionali.

Questa ipotesi di distribuzione del significato può essere facilmente immaginata come una teoria caratterizzata da una forte connotazione operazionale. Tuttavia, essa risulta carente nella definizione dei suoi principi se paragonata agli approcci descritti nel capitolo precedente. La differenziazione semantica è espressa infatti in termine di variazione delle distribuzioni, senza specificare né cosa sia l'informazione di distribuzione, né quale genere di differenze essa rappresenta nel significato. Nel tentativo di esplicitare comunque alcuni dettagli della differenze è allora necessario introdurre alcuni aspetti del significato così da poter meglio comprendere la natura di tale differenziazione semantica.

L'analisi funzionale è forse l'approccio più semplice per differenziare i segni di una lingua. Saussure propose di distinguere le relazioni funzionali in due categorie principali: sintagmatiche e paradigmatiche. Le prime riguardano il posizionamento delle parole relazionando quest'ultime in base alle loro co-occorrenze nel testo. Le relazioni sintagmatiche sono quindi relazioni che esprimono la combinatoria delle parole e delle frase denominati *sintagmi*. Le relazioni paradigmatiche sono invece rivolte a descrivere sostituzioni e evidenziano entità linguistiche che non compaiono nello stesso testo ma che comunque sono presenti negli stessi contesti. Esempi di possibili sostituti possono essere le parole *brilliant* e *opaque* nel contesto dei colori. Queste relazioni possono quindi esprimere la possibilità che la scelta di un'entità linguistica escluda la presenza di altre entità ad essa paradigmaticamente collegate.

Aspetto distintivo di questo approccio rispetto agli approcci trattati nei capitoli precedenti è la chiara posizione che assegna la supremazia della rappresentazione distribuzionale rispetto alle altre caratterizzazioni ontologiche dell'essenza del significato. In questo contesto tutte le problematiche derivanti dal riferimento a qualche modello esterno che rappresenti il mondo scompaiono.

Secondo la metodologia distribuzionale ciò che definisce il significato sono asserzioni su come le entità linguistiche di base sono collegate fra di loro formando una certa distribuzione. Differenze di significato si traducono in differenze nelle distribuzioni associate abilitando una misura quantitativa nella differenza fra diverse entità linguistiche. In questo contesto l'unico aspetto importante riguarda la procedura utilizzata per stabilire la similarità semantica e le informazioni di contesto.

Nella *Latent Semantic Analysis*(LSA), ad esempio, i contesti sono definiti su un universo definito dal documento(reminiscenza questa della tradizione di estrazione della conoscenza); mentre nei modelli più recenti basati su reti neurali, discussi nella prossima sezione, i contesti sono definiti dalle parole in un predeterminato intorno. Ovviamente una tale differenza metodologica produce una diversa tipologia di similarità semantica: la prima tipologia di modelli cattura relazioni più ampie e quindi *contestuali* o di argomento, come

car-tyres, mentre i secondi focalizzano la loro attenzione su relazioni incentrate sul singolo termine, ad esempio *car-vehicle*.

Tipicamente l'approccio utilizzato per costruire i modelli distribuzionali si basa sul conteggio delle co-occorrenze delle parole. Secondo tale approccio le caratteristiche di una parola sono espresse attraverso l'utilizzo di altre parole costruendo di fatto matrici di occorrenza o di contesto. Su queste informazioni sono poi applicate le tecniche per l'apprendimento dei modelli rappresentanti le parole. Fra queste, il *Singular Value Decomposition* è forse una delle tecniche più utilizzate in LSA. Attraverso SVD è possibile fattorizzare la matrice delle occorrenze parole-contesto M nel prodotto di tre matrici: due ortonormali e una matrice diagonale caratterizzata dalla presenza degli autovalori di M in ordine decrescente sulla diagonale principale. La matrice M è comunemente costruita sulla base della *Positive Pointwise Mutual Information* che è una misura del livello di associazione fra due parole definita in termini di probabilità congiunta in relazione alla loro probabilità marginale, $PPMI(w_1, w_2) = max(0, P(w_1, w_2)/(P(w_1)P(w_1)))$.

Tuttavia, recentemente l'apprendimento delle rappresentazioni delle parole è quasi esclusivamente eseguito con reti neurali. Questi modelli del linguaggio hanno raggiunto in poco tempo un enorme consenso nella comunità della linguistica computazionale. Il motivo di tale successo sarà argomento della prossima sezione.

3.2 Modelli per il linguaggio e reti neurali

Nelle recenti architetture dei sistemi per l'elaborazione del linguaggio naturale, le rappresentazioni vettoriali di parole apprese utilizzando reti neurali hanno completamente sostituito i modelli costruiti con le tradizionale caratteristiche distribuzionali. In questi modelli le parole sono rappresentate all'interno di uno spazio vettoriale (la cui dimensionalità varia da 300 a 800) da un vettore loro associato, denominato *word embedding* o vettore di parola.

I vettori di parole sono attualmente appresi attraverso l'impiego di reti neurali complesse su enormi collezioni di testo non annotato. Negli ultimi anni, grazie ai recenti avanzamenti tecnologici(soprattutto in termini di capacità computazionale tensoriale a costi relativamente contenuti) si è assistito all'esplosione di modelli costruiti con sofisticate reti neurali. Generalmente parlando, costruire vettori di parole per un vocabolario ampio utilizzando una rete neurale complessa è un compito computazionalmente costoso. Ciò motiva l'assenza di modelli addestrati con questa metodologia nel recente passato.

I vettori di parole sono molto vicini ai modelli del linguaggio. Questi modelli provano principalmente a definire la probabilità di una parola w_i come condizionata dalle precedenti $n-1$ parole $p(w_i|w_{i-1}, \ldots, w_{i-n+1})$. Applicando la regola di costruzione a catena, sotto l'assunzione della proprietà di Markov del processo, è possibile approssimare il prodotto dell'intera frase o del documento con il prodotto delle singole probabilità riferite alle parole e alle n

parole che la precedono come:

$$p(w_1, \ldots, w_T) = \prod_i p(w_i | w_{i-1}, \ldots, w_{i-n+1})$$

Nei modelli per il linguaggio basati su *n-gram*, ad esempio, la probabilità di una parola può essere calcolata basandosi sulla frequenza dei suoi costituenti *n-gram*:

$$p(w_i | w_{i-1}, \ldots, w_{i-n+1}) = count(w_{i-n+1}, \ldots, w_{i-1}, w_i)$$

In questo modo risulta banale definire la probabilità di un bigramma o del più famoso *5-gram* spesso utilizzato come modello di riferimento per l'analisi delle prestazioni.

Nel contesto delle reti neurali quanto appena detto può essere realizzato definendo un semplice classificatore *softmax* come livello finale della rete:

$$p(w_i | w_{i-1}, \ldots, w_{i-n+1}) = \frac{exp(h^T v'_{w_i})}{\sum_{w_t \in V} exp(h^T v'_{w_t})} \tag{3.1}$$

Qui il prodotto scalare $h^T v'_{w_i}$ calcola la log-probabilità non normalizzata di w_i, mentre il denominatore è il fattore di normalizzazione calcolato sull'intero vocabolario. Il vettore h è il vettore prodotto dal penultimo livello della rete, mentre v'_{w_i} è il vettore di uscita della rete.

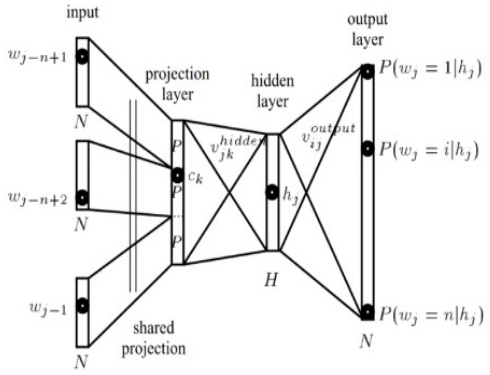


Figura 3.1. Il modello neurale per il linguaggio proposto da Bengio in [6].

Si noti qui che il vettore v'_{w_i}, pur rappresentando la parola w_i, è appreso in modo indipendente dal vettore d'ingresso v_{w_i}. Soprattutto si noti come sia necessario calcolare la probabilità per ogni parola w_i nel vocabolario. Sebbene costoso in termini computazionale, ciò può essere realizzato come semplice moltiplicazione fra il vettore h e la matrice dei pesi le cui righe si riferiscono ai singoli vettori v'_{w_i} per ogni w_i nel vocabolario.

Nella sua prima definizione di modello neurale per il linguaggio, Bengio propose di impiegare una rete neurale con un solo livello intermedio. L'obiettivo di tale modello era predire la prossima parola data una sequenza. Una sua rappresentazione è mostrata in figura 3.2.

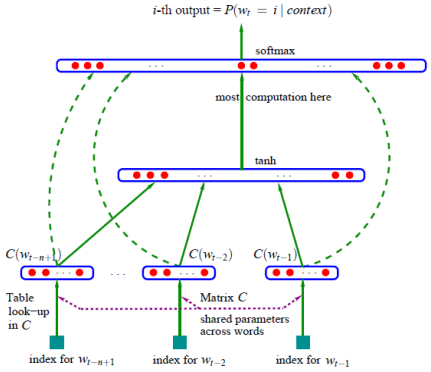


Figura 3.2. Modello neurale per il linguaggio proposto da Bengio in [6].

L'architettura in 3.2 rappresenta il prototipo a partire dal quale i più moderni approcci sono costruiti. E' possibile infatti trovare all'interno di questi ultimi i blocchi generali qui definiti:

- *Embedding Layer*: è il livello che genera i vettori di parole moltiplicando un vettore di indici con la matrice contenete tutti i vettori;
- *Intermediate Layer*: uno o più livelli intermedi che producono rappresentazioni del vettore d'ingresso;
- *Classification Layer*: è il livello finale che produce una distribuzione di probabilità su ogni parola del vocabolario.

Addestrare il modello ha come obiettivo la massimizzazione della seguente funzione oggetto:

$$J_\theta = \frac{1}{T} \sum_{t=1}^{T} log\ p(w_t | w_{t-1}, \ldots, w_{t-n+1}) \tag{3.2}$$

Durante l'apprendimento quindi si tenta di massimizzare la media delle log-probabilità di tutte le parole dell'intero vocabolario assumendo date n precedenti parole.

Tuttavia, come osservato da Bengio, il livello finale basato sul classificatore softmax rappresenta il punto critico del sistema. Calcolare tale valore ha infatti un costo proporzionale al numero di parole nel vocabolario, che tipicamente si attesta sull'ordine di qualche centinaio di migliaia. Trovare quindi un modo per mitigare questo costo è al centro delle successive ricerche sui modelli neurali per il linguaggio.

3.3 Il modello proposto da Collobert e Weston

Inizialmente la limitata capacità computazionale e l'assenza di algoritmi veramente efficienti non permettevano l'addestramento di modelli neurali su vocabolari estesi. Sin dai primi lavori di Bengio sui modelli neurali apparve infatti evidente l'enorme costo computazionale richiesto durante la fase di addestramento. Ciononostante, occorsero alcuni anni di ricerca per poter definire approcci alternativi capaci di evitare il costoso calcolo nel livello finale della rete.

Collobert e Weston furono fra i primi a mostrare un approccio alternativo per addestrare un modello contente utili informazioni sintattiche e semantiche su un vocabolario relativamente ampio. La soluzione proposta prevedeva l'impiego di una differente funzione oggetto: invece di basarsi su criteri di entropia incrociata come fatto da Bengio, la rete era addestrata così da assegnare un punteggio alto alle sequenze di parole corrette. Per fare ciò gli autori impiegavano la seguente funzione:

$$J_\theta = \sum_{x \in X} \sum_{w \in V} max\{0, 1 - f_\theta(x) + f_\theta(x^w)\} \tag{3.3}$$

Le sequenze corrette erano estratte dalla collezione di testi impiegando una finestra x contenente n. Per ogni finestra, si genera una sua versione *corrotta*, x^w, sostituendo la parola nel centro della finestra con una parola w estratta da V.

La fase di addestramento quindi massimizza la distanza fra i punteggi assegnati dal modello per le sequenze corrette e incorrette. Sebbene diverso dal precedente approccio, questo modello genera vettori di parole con caratteristiche comune agli altri modelli. Ad esempio non solo parole riferite allo stesso concetto sono raggruppate insieme, ma anche parole sintatticamente vicine risiedono all'interno di piccole porzioni dello spazio vettoriale.

Tuttavia, se da un lato Collobert e Weston eliminano il costo relativo al calcolo della funzione softmax, dall'altro essi mantengono una struttura interna alla rete totalmente complessa. Quest'ultima è risultato essere in termini computazionali un'ulteriore criticità del sistema[1], risolta solo successivamente dal modello word2vec.

3.4 Il modello *Word2Vec*

Nel 2013 Mikolov e colleghi proposero il modello word2vec che rapidamente diventò il più diffuso modello neurale per la costruzione di vettori di parole. La strategia adottata si basa sulla definizione di due architetture: Continuous

[1] Gli autori dichiararono che occorsero sette settimane per l'addestramento del modello su un vocabolario di 130000 termini.

bag-of-words (CBOW), e Skip-gram. In entrambe l'obiettivo comune è evitare le costose operazione richieste nei modelli precedenti. Rispetto a questi, infatti, word2vec non fa uso né di operazioni non lineari né di costosi livelli intermedi. Al contrario esso sfrutta diverse strategie per velocizzare il processo di apprendimento e per incrementare i livelli di accuratezza.

Continuous bag-of-words

I modelli discussi finora sono stati tutti focalizzati sulla capacità di predire la parola successiva in una sequenza di n parole ricevuta come ingresso del sistema. Tale tendenza è giustificata principalmente dalle abilità predittive richieste al modello in fase di valutazione. Ma come fatto notare da Mikolov, restringendo lo scopo del modello alla sola produzione di accurati vettori di parole è possibile rilassare alcune restrizioni imposte sui precedenti modelli.

In CBOW, infatti, per ogni parola si costruisce un contesto utilizzando le n parole che precedono e seguono una determinata parola *target* w_t. Tale contesto è definito come una rappresentazione continua per la parola in esame, ed è mostrata in figura 3.3.

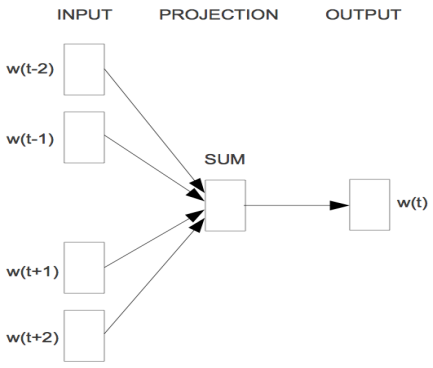


Figura 3.3. Contesto di parole per una determinata parola w_t in CBOW.

La funzione obiettivo è leggermente differente dalla funzione impiegata precedentemente. Ovvero invece di ricevere le n parole precedenti essa riceve una finestra di parole che circondano la parola target w_t:

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} log\ p(w_t|w_{t-n},\ldots,w_{t-1},w_{t+1},\ldots,w_{t+n}) \tag{3.4}$$

3.4.1 Il modello *Skip-Gram*

Nel modello *skip-gram* ogni parola ed ogni contesto è invece rappresentato da un vettore di dimensione fissata. Definiti con W e C rispettivamente l'insieme

delle parole e dei contesti, sia d la dimensionalità scelta per lo spazio vettoriale, allora:

$$\forall c \in W \exists c \in C : v_w, v_c \in R^d, (v_w \cdot v_c) \in P$$

dove P è l'insieme delle coppie parole-contesto il cui prodotto vettoriale sia stato massimizzato. Gli elementi dei vettori sono le incognite del modello è devono essere appresi attraverso la fase di addestramento.

La procedura d'addestramento denominata *negative-sampling*(SNGS) assume un insieme D di coppie (w, c) con w e c provenienti da una ampia collezione di testi. Si denoti adesso la probabilità che una data coppia (w, c) provenga effettivamente dai dati con $p(D = 1|w, c)$ e con $p(D = 0|w, c) = 1 - p(D = 1|w, c)$ il caso contrario. La distribuzione del modello è

$$p(D = 1|w, c) = \frac{1}{1 + e^{v_w \cdot v_c}} \tag{3.5}$$

L'addestramento allora procede tentando di massimizzare la *log*-probabilità per le coppie osservate. Da qui la definizione della seguente funzione obiettivo:

$$argmax_{v_w \cdot v_c} \sum_{(w,c) \in D} log \frac{1}{1 + e^{v_w \cdot v_c}} \tag{3.6}$$

Tuttavia, ponendo $v_w = v_c$ e $v_w \cdot v_c = K$ per valori di K molto grandi tale funzione conduce ad una soluzione banale $p(D = 1|w, c) = 1$. Per prevenire questo situazione si crea artificialmente un secondo insieme D' e si assume che tutte le coppie generate casualmente siano incorrette. Eseguito l'addestramento sui due insiemi il risultato ottenuto sarà una collezione di vettori che risulteranno simili per coppie osservate nei dati.

Si noti infine come in questo approccio vi sia un'implicita costrizione operata sul modello. Poiché si utilizza come contesto l'intorno di una parole, ne consegue che gli elementi di W e C siano simili. Soprattutto saranno simili i sistemi di tipi utilizzati per entrambi. Come osservato da Levy e Goldberg ciò non è necessariamente un vincolo imposto dal modello. Invece esso deriva dall'applicazione della tipica metodologia denominata *Linear Bag-of-Words Contexts* che definisce un intorno lineare di larghezza $2k$ per ogni parola in esame. Ad esempio, per l'espressione $w_0 w_1 w_2 w_3 w_4$ considerando come parola d'interesse w_2 si genera come suo contesto associato $\{w_0, w_1, w_3, w_4\}$.

Come già descritto precedentemente, intorni per valori di k vicini a 5 sono più adatti a catturare informazioni generali, mentre valori di k inferiori sono rivolti ad informazioni più specifiche della parola corrente.

Nel tentativo di costruire il contesto in modo tale che esso fornisca qualche più utile informazione, Levy e Goldberg generalizzano il modello skip-gram utilizzando come contesti l'insieme di dipendenze di ogni parola.

3.5 Contesti di dipendenza e sostituzione lessicale

Come descritto nei capitoli precedenti, i parser basati sulle grammatiche delle dipendenze offrono attualmente un'alta affidabilità e tempi di elaborazione sufficientemente bassi. Ciò permette la loro applicazione anche su grandi collezioni di testi. Utilizzando un parser per l'analisi delle dipendenze, Levy e Goldberg propongono di costruire il contesto relativo ad un parola d'interesse a partire dall'albero delle dipendenze[2] prodotto attraverso l'elaborazione dell'espressione sotto esame.

Utilizzando il contesto definito attraverso le relazioni di dipendenza si evidenziano aspetti *funzionali* che sono preclusi all'approccio basato su *bag-of-word*. Infatti, da un lato, il contesto generato dalle dipendenze permette di catturare relazioni che di solito si estendono oltre la copertura delle finestre tipicamente impiegate, dall'altro eliminano qualche informazione che non è direttamente pertinente col il termine in esame. Soprattutto le dipendenze permettono di caratterizzare individualmente i singoli elementi del contesto; è infatti abbastanza immediato arricchire gli elementi del contesto con le relazioni di dipendenza con le quali sono legati alla parola d'interesse. Il tipo di similarità che un tale contesto riesce a catturare è più incentrato su aspetti funzionali che di tema o d'argomento.

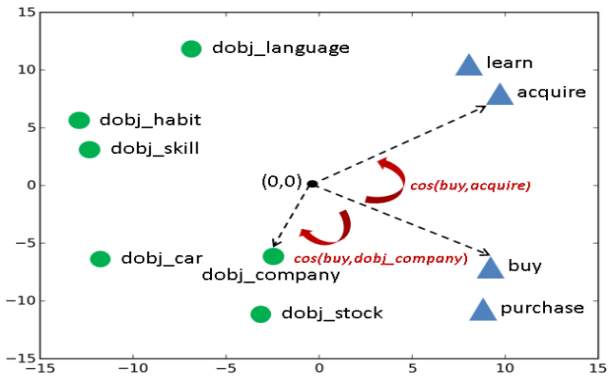


Figura 3.4. Esempio di identificazione dei sostituti utilizzando il modello proposto da Melamud e Levy mostrato in uno spazio ridotto a 2-dimensioni. Per la parola d'interesse *acquire* con contesto sintatticamente annotato "*dobj_company*", anche se *learn* è la parola più vicina nello spazio si seleziona la parola *buy*. Quest'ultima risulta essere un miglior candidato per la sostituzione poiché è la più vicina ad entrambi i riferimenti *acquire* e *dobj_company*.

Basandosi su questo modello di skip-gram generalizzato Melamud e Levy proposero successivamente un approccio per definire potenziali sostituti

[2] Gli autori utilizzano la versione *collassata* dell'albero, cioè con etichette delle dipendenza estese con informazioni sulle preposizione.

di termini all'interno di un contesto sintatticamente annotato. Il problema di predire il giusto sostituto per un termine in un contesto ha vissuto negli ultimi anni un incremento di popolarità. Per definizione questo problema richiede al modello di predire potenziali sostituti per una data parola sotto la condizione di preservare il suo significato nel contesto dato. Gli autori proposero alcune misure, mostrate in tabella 3.1, per definire un indice di sostituibilità sensibile al contesto. Tale misure nascono dall'assunzione che potenziali candidati, adatti alla sostituzione, debbano essere semanticamente simili, o detto in altri termini funzionalmente compatibili con il contesto. Utilizzando una misura di similarità del secondo ordine, fra la parola d'interesse ed un suo potenziale sostituto, e una misura di similarità del primo ordine per valutare la compatibilità con il contesto, si ottiene una misura di similarità semantica maggiormente incentrata su aspetti funzionali che su caratterizzazioni statiche del significato dei due termini da comparare. Un esempio è mostrato in figura 3.4.

Tabella 3.1. Metriche di sostituibilità considerate dal modello

Nome	
Add	$\frac{cos(s,t)+\sum_{c\in C} cos(s,c)}{\lvert C\rvert+1}$
BalAdd	$\frac{\lvert C\rvert \cdot cos(s,t)+\sum_{c\in C} cos(s,c)}{2\cdot\lvert C\rvert}$
Mult	$\sqrt[\lvert C\rvert+1]{pcos(s,t)\cdot\prod_{c\in C} pcos(s,c)}$
BalMult	$\sqrt[2\cdot\lvert C\rvert]{pcos(s,t)^{\lvert C\rvert}\cdot\prod_{c\in C} pcos(s,c)}$

Come mostrato in tabella 3.1, per combinare gli *score* dei singoli elementi le operazioni *Add* e $BalAdd$ si basano su una media aritmetica, mentre invece $Mult$ e $BalMult$ si basano su una media geometrica.

Descrivere la *ratio* dietro queste operazioni è abbastanza semplice. La combinazione additiva operata nelle prime due operazioni permette di ottenere un misura meno rigida, permettendo quindi alti valori anche in presenza di elementi il cui contributo è zero. Mentre nelle ultime due operazioni, si rappresenta invece un vincolo di una più stretta compatibilità. In queste misure si richiede infatti un alto grado di similarità per tutti gli elementi; un potenziale sostituto deve quindi avere un'alta similarità con tutti gli elemento del contesto oltre che con la parola d'interesse.

Altro aspetto da evidenziare in queste operazioni proposte è il diverso trattamento dato al contributo del termine relativo alla compatibilità con il contesto. Mentre in *Add* e $Mult$ il peso di questo termine potrebbe prevalere sulla similarità sulle parole, se si considera un contesto molto ricco di elementi, in $BalAdd$ e $BalMult$ si tenta di rimediare a questa possibilità sotto l'ipotesi che il contributo della compatibilità contestuale dovrebbe essere fissa e non dipendente dagli elementi costituenti il contesto.

3.6 Glove

Ultimo esempio discusso in questo capitolo è il modello proposto da Pennington e colleghi denominato *Glove*. Lo scopo di tale modello è rendere esplicito ciò che in SNGS è lasciato implicito. Gli autori di Glove partono dalla convinzione che il rapporto fra le probabilità di co-occorrere di due parole sia ciò che effettivamente contiene informazione. Da questa idea costruiscono un modello capace di catturare questa informazione all'interno dei suoi vettori.

Glove è progettato così da ricavare attraverso la differenza dei suoi vettori il significato derivante dalla giustapposizione di due parole. Per esempio, il concetto che distingue *man* da *woman*, può essere specificato allo stesso modo da differenti coppie di parole come *king* e *queen*. La principale intuizione in Glove è che il rapporto fra le probabilità di co-occorrere di due parole abbia le potenzialità per catturare questa forma di significato.

Si consideri le probabilità di co-occorrenza delle parole *ice* e *steam* con alcune parole presenti nel vocabolario:

Tabella 3.2. Elenco delle dipendenze universali

Probabilità e Ratio	*k=solid*	*k=gas*	*k=water*	*k=fashion*
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-3}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Come è facile aspettarsi, la frequenza di co-occorrenze del termine *ice* è più alta con il termine *solid* che con il termine *gas*. Allo stesso modo è ragionevole assumere che per *steam* sia vero il contrario. Altro aspetto evidenziato dalla tabella è una certa frequenza con il termine *water*, e una bassa co-occorrenza con il termine *fashion*.

E' importante notare come solo all'interno nell'ultima riga, riferita appunto al rapporto fra le probabilità, il rumore introdotto dalle parole non-discriminative è rimosso. In questa riga i valori superiori all'unità indicano proprietà specifiche o comunque riconducibili al numeratore, mentre quelle inferiori sono da attribuirsi al denominatore.

Secondo gli autori nel caso appena riportato il rapporto fra le probabilità codifica grossolanamente una forma di significato riconducibile al concetto astratto della fase termodinamica.

La funzione utilizzata da Glove è una funzione pesata dei minimi quadrati che è rivolta a minimizzare le differenze fra i prodotti scalari e il logaritmo del numero delle loro co-occorrenze.

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \hat{w}_j + b_i + \hat{b}_j - log\ X_{ij})^2 \tag{3.7}$$

In J, w_i e b_i sono rispettivamente il vettore ed il *bias* riferiti alla parola i, mentre $\hat{w}_j$ e $\hat{b}_j$ sono il vettore di contesto di j e il suo relativo bias. X_{ij} è il numero di volte che le parole i e j risiedono nella stessa finestra, e f è una funzione di peso che assegna un valore ridotto alle co-occorrenze troppo rare o frequenti.

3.7 Sommario e ulteriori riferimenti

I primi riferimenti sulla semantica distribuzionale sono i lavori di Harris [61], di Firth [45] e Wittgenstein [147]. Tuttavia, solo negli ultimi venti anni si è assistito alla diffusione di metodi per derivare automaticamente misure di similarità semantica fra termini in una collezione di testi. Fra i primi modelli per rappresentare i termini in uno spazio multi-dimensionale sono l' Hyperspace Analogue to Language(HAL) proposto da Lund e Burgess in [94], lo spazio di parole di Schutze in [134], ed il Latent Semantic Analysis (LSA) di Landauer e Dumais in [81]. Successivamente furono presentati modelli probabilistici come il Probabilistic LSA di Hofman in [64], il Latent Dirichlet Allocation di Blei e colleghi in [9], ed i Probabilistic Topics Models di Griffiths e Steyvers in [56]. Ovviamente entrambi gli approcci permettono la stima di similarità semantica fra termini. Ma mentre i primi comparano i termini usando metriche di distanza in spazi vettoriali multi-dimensionali, i secondi misurano la similarità in base al grado di verosimiglianza della distribuzione di argomento. Nei modelli probabilistici infatti i documento sono visti come una combinazione di argomenti. I termini al loro interno sono rappresentati in base alla probabilità di essere incontrati durante la discussione su uno specifico argomento.

Il lavoro di Bengio e colleghi [5] definisce le fondamenta dei modelli per il linguaggio naturale basati sull'utilizzo delle reti neurali. Tuttavia, solo alcuni anni dopo, con il modello sviluppato da Collobert e Weston, e presentato in [27], si poté iniziare ad apprezzare le potenzialità dell'approccio delineato da Bengio. La conferma definitiva arrivò qualche anno dopo con il modello word2vec proposto Mikolov in [102, 103]. Uno studio approfondito di alcuni aspetti di questo modello, tralasciati dagli autori originali, è fornito nel lavoro di Goldberg e Levy [54]; mentre una discussione più generale sulle misure di similarità basate sulla semantica distribuzionale è fornita in [88]. Sempre Goldberg e Levy sono gli autori dell'approccio, pubblicato in [87], per la costruzione dei contesti basato sulle relazioni di dipendenza. Infine Melamud e colleghi presentano in [101] un'efficace tecnica per la sostituzioni di termini in un contesto sintatticamente annotato. L'ultimo modello discusso in questo capitolo, Glove, è presentato da Pennington e colleghi in [69].

4

Risorse lessicali e ontologie

Le principali risorse lessicali e semantiche considerate in questo lavoro ricadono in due principali macro categorie. Nella prima gli elementi principali attorno ai quali costruire la conoscenza sono le parole. Questa tipologia è rivolta a creare ciò che oggigiorno è identificato con il termine *risorse semantico-lessicali*. Il loro scopo è tener traccia di informazione contenuta in enormi collezioni di testi la cui natura è prevalentemente cognitivo-linguistica. La seconda categoria segue invece una metodologia più formale, attraverso la definizione di teorie logiche la cui consistenza è fondata su determinate primitive concettuali. Esempi di quest'ultimo tipo sono le *ontologie*.

In questo capitolo saranno discussi alcuni esempi di applicazione per entrambe le categorie, evidenziandone non solo i princìpi fondanti, ma anche le principali differenze trai vari approcci, anche all'interno di una stessa categoria.

4.1 Risorse lessico-semantiche

Le risorse lessicali mantengono informazione sull'organizzazione semantica del lessico di un linguaggio. L'idea di partenza riguarda l'uso dei termini di un particolare linguaggio che permette di esplicitare un certo grado di regolarità semantica. Quest'ultimo a sua volta può essere impiegato per catturare relazioni semantico-lessicali.

Le relazioni estratte costruite a partire dalle parti del discorso(*part-of-speech*), dalla condivisione di alcune caratteristiche o dalla co-occorrenza all'interno di piccole espressioni, sono suddivise in due categorie principali: paradigmatiche e sintagmatiche. Fra le relazioni paradigmatiche le più importanti sono sicuramente quelle dei sinonimi, dell'iperonimia e della meronimia. I sinonimi possono essere definiti come parole differenti nella forma ma simili nel significato che possono essere sostituiti, all'interno di un'espressione, senza che ne sia modificato il significato ad essa ascritto. Un'ulteriore definizione

di sinonimo, proposta da Miller e Fellbaum, circoscrive la relazione come appartenente ad un determinato contesto; cioè la nozione di sostituzione, come conservazione del valore di verità, può essere definita solo in riferimento ad un determinato contesto.

L'iperonimia di un termine A lo lega ad termine B che generalizza, secondo una qualche concettualizzazione, l'idea stessa di A. Se da un lato A eredita tutte le proprietà da B, dall'altro A deve necessariamente aggiungere qualche caratteristica che lo contraddistingue. Ciò può essere espresso in termini di implicazione come:

$$\forall A \in Hyp(B) \exists C : A \rightarrow C, C \notin Hyp(B)$$

La possibilità di percorrere questi legami in entrambi i versi, cioè generalizzando e specificando le proprietà di un termine, ha reso le strutture generate attraverso questa relazione elemento fondamentale di molte risorse lessicali.

La meronimia, infine, modella la relazione di costituzione fra un *tutto* e le sue *parti*. La meronimia realizzata come una collezione di relazioni può caratterizzare ulteriormente ogni parte con la specifica funzione che la parte in questione svolge all'interno del particolare tutto. A titolo esemplificativo in tabella 4.1 sono riportate sei tipologie di relazioni proposte da Winston accompagnate dai relativi esempi.

Relazione	Esempio
component–integral object	leg–table
member–collection	soldier–army
stuff–object	alcohol–wine
feature-activity	pay–shop
place–area	oasis– desert
portion–mass	meter–kilometer

Tabella 4.1. Tipologie di relazioni di meronimia proposte Winston.

Capita a volte che nelle risorse lessicali sia presente un'ulteriore relazione che identifica l' "opposto" di un termine. Nell'accezione più utilizzata questa relazione non implica una completa differenziazione fra due termini, bensì ne riduce la portata ad un solo elemento distintivo. Da ciò si può definire la relazione di opposizione[1] come:

$$\forall A \in Op(B) \exists C : A \rightarrow C, B \rightarrow \neg C$$

Per quel che riguarda le relazioni sintagmatiche invece esse si basano sulla "forza" che una determinata associazione di parole ha in un particolare lin-

[1] Spesso capita di associare l'idea di antonimia con la definizione di opposizione. E' tuttavia opportuno restringere tale equivalenza ai casi in cui l'opposizione sia *misurabile* in qualche modo.

guaggio. Si prendano come esempio le espressione ideomatiche. Esse rappresentano rigide combinazioni di parole che sono comprese dagli utilizzatori del linguaggio come un unico elemento, come "*to kick the bucket*". Mentre espressioni molto frequenti, tipo *book-read*, sono sempre relazioni sintagmatiche ma rappresentano semplicemente co-occorrenze di termini in un predeterminato intorno.

All'interno delle risorse lessicali questi fenomeni sono rappresentati in modi differenti: infatti se da un lato gli idiomi sono associati ad un singolo elemento, lo stesso non può essere detto per le co-occorrenze che sono invece rappresentate come semplici riferimenti fra gli elementi coinvolti.

Prima di introdurre gli esempi di risorse semantiche è opportuno sottolineare come queste risorse non sia vincolate da una definizione formale. Il loro utilizzo non si basa quindi sull'assunzione che l'inferenza offerta sia sempre corretta: molto spesso infatti mancano gli assiomi per determinare la correttezza strutturale della base di conoscenza. In WordNet, ad esempio, la relazione di iponimia descrive un ordinamento di subordinazione basato sulla restrizione del significato. Per costruzione, tale relazione di iponimia prevede il passaggio delle caratteristiche dal termine più generale verso quello più specifico. Se questa regola è valida in generale, esistono delle eccezioni come nel caso di "*wheelchair*". Quest'ultimo infatti come iponimo di "*chair*" eredita tutte le proprietà di quest'ultimo. Ma "*chair*" a sua volta come iponimo di "*furniture*" è caratterizzato anche dalle sue caratteristiche. Da qui l'incongruenza che il sistema è portato ad asserire affermando che "*wheelchair*" sia una elemento d'arredo. Un tale problematica, tuttavia, non deriva dal sistema in sé, ma da come la conoscenza del mondo(o generica) è organizzata al suo interno.

4.2 Esempi di risorse lessicali

4.2.1 WordNet

WordNet struttura la conoscenza lessico-semantica di un linguaggio in una rete di collegamenti. Sviluppato presso l'università di Princeton, esso rappresenta ancora oggi una delle risorse più utilizzate nei sistemi per la comprensione del linguaggio naturale. Fra la varie applicazioni WordNet è soprattutto impiegato nella risoluzione del senso di parole polisemiche.

Il motivo di quest'ampia diffusione è da attribuire principalmente all'ampia copertura di termini, circa 160.000 per la lingua inglese, che questa risorsa possiede. I lemmi al suo interno, suddivisi su quattro categorie sintattiche (nomi, verbi, aggettivi ed avverbi), sono organizzati in gruppi di sinonimi chiamati synset che costituiscono gli elementi fondamentali della rete.

I synset sono spesso considerati equivalenti alla nozione di concetto, ma tale definizione può essere fuorviante alla luce delle questioni che verranno introdotte successivamente parlando delle ontologie. I synset possono essere

meglio caratterizzati dal loro obiettivo che prevede l'esplicitazione di un significato attraverso un insieme di lemmi che lo richiamano. Ad esempio il synset per *dog* è legato ai synset:

$$\{dog,\ domestic\ dog,\ Canis\ familiaris\}$$

Synset e parole in esse contenute sono collegate attraverso le relazioni paradigmatiche introdotte sopra. In quest'esempio attraverso la relazione d'iperonimia si arriva al synset {*domestic animal, domesticated animal*}.

Ogni synset è ulteriormente caratterizzato da una *gloss*, cioè da una o più frasi che forniscono una definizione utile a spiegare il significato associato al synset.

Attualmente WordNet contiene un grafo molto denso di relazioni definite fra i synset che appartengono alla stessa parte del discorso. Tuttavia il numero di relazioni fra parti del discorso differenti è decisamente inferiore. WordNet può quindi essere visto come quattro distinte reti, una per ognuna delle categorie sintattiche. Ciò rende WordNet una risorsa costruita con una predilezione verso gli aspetti linguistici rispetto quelli cognitivi. Ciononostante, attraverso le relazioni attributo-valore(*noun-adj*:complexity-complex) e di derivazione(*verb-noun*:construct-construction) è possibile collegare synset di parti del discorso differenti.

Nonostante la sua ampia diffusione e copertura, WordNet non è esente da critiche. Le due principali riguardano la sua stessa architettura: da un lato si critica la forse troppo rigida distinzione fra i sensi delle parole, dall'altro l'assenza di una garanzia sulla consistenza concettuale all'interno della rete. Sebbene queste critiche trovino consenso all'interno della comunità dei linguisti computazionali, esse non hanno limitato più di tanto l'adozione di WordNet in un gran numero di applicazioni concernenti il linguaggio naturale.

4.2.2 FrameNet

FrameNet descrive la conoscenza riconducibile a determinate situazioni attraverso l'approccio delineato da Fillmore nella semantica dei *frame*. Ruppenhofer descrive lo scopo di FrameNet come la volontà di documentare attraverso evidenze, riscontrate in una collezione di testi, le possibili combinazioni(di natura sintattica e semantica) che una parola può assumere in un determinato contesto, o frame. Soprattutto, l'analisi svolta da FrameNet è mirata all'esplicitazione dell'assegnamento dei ruoli semantici. Analisi questa dei ruoli semantici che non si limita ai soli verbi ma è estesa anche a nomi, aggettivi, avverbi e preposizioni.

In questo contesto l'elemento principale è il frame. Un frame descrive una situazione prototipale esprimibile in linguaggio naturale, ed è caratterizzato da un insieme di ruoli semantici che ne descrive i partecipanti. Questi ruoli, a differenza dei ruoli descritti in altre risorse lessicali, prendono valore solo se riferiti ad un frame e solo al suo interno possono trovare una concreta realizzazione. Si consideri, ad esempio, la situazione riferita al verbo "*to give*",

il suo frame di riferimento è *GIVING* con i seguenti ruoli da attribuire alle realizzazioni nel linguaggio naturale:

$$\{DONOR, RECIPIENT, THEME\}$$

Nell'espressione *"Donald gave Hilary a job"*, l'annotazione fornita da FrameNet darà a Donald il ruolo di DONOR, a Hilary quello di RECIPIENT, e job sarà identificato con THEME.

L'esempio appena mostrato evidenzia una caratteristica molto importante di FrameNet: ovvero attraverso un sempre crescente numero di frasi annotate FrameNet fornisce informazioni sulle realizzazioni sintattiche degli elementi del frame. Nel caso d'esempio il ruolo RECIPIENT è molto spesso rappresentato o da una frase nominale che funge da oggetto indiretto o da una frase preposizionale introdotta dalla preposizione *to*. Ma aldilà delle informazioni sintattiche le parole stesse giocano un ruolo importante all'interno di FrameNet.

Con il termine *lexical unit* si fa riferimento ad un termine che evoca uno specifico frame. Una volta identificata un'unità lessicale è possibile risalire al frame o ai frame ai quali quella particolare parola è associata. Successivamente attraverso l'annotazione è auspicabile immaginare che il processo di identificazione del ruolo, o dei ruoli, che quel termine può svolgere nella situazione sia facilitato.

Infine FrameNet definisce un insieme di relazioni che possono sussistere fra i frame la cui lista completa è riportata di seguito:

- *Inheritance* : quando un frame specifica aspetti di un frame più astratto. Questa relazione implica che tutto ciò che è vero per il frame genitore rimane vero per il frame che lo specifica, e che esiste un allineamento fra i ruoli semantici dei due frame.
- *Perspectivized_in* : serve a collegare un frame, che ha una prospettiva neutrale nel descrivere una situazione, ai relativi frame che invece rappresentano la stessa situazione ma da punti di vista differenti. Un classico esempio è il frame *Commerce_transfer-goods*, e i relativi frame *Commerce_sell* e *Commerce_buy*.
- *Subframe* : usato per suddividere situazioni complesse in un insieme di situazioni più specifiche lungo qualche aspetto. Il frame *Criminal_process*, ad esempio, descrive uno scenario complesso che è costruito all'interno di FrameNet attraverso i frame *Arrest*, *Trial*, e così via.
- *Precedes* : la costruzione di uno scenario complesso può richiede un ordinamento temporale dei suoi sotto-frame. Questa relazione serve a definire tale ordine.
- *Causative_of* e *Inchoative_of* : servono per definire relazioni fra descrizioni di stato e descrizioni causali o incoative, tipo il frame *Position_on_a_scale* può essere collegato tramite queste relazioni a *Cause_change_of_scalar_position* e *Change_position_on_a_scale*.

- *Using* : due frame sono legati da questa relazione se uno dei due coinvolge in qualche modo l'altro. Si usa questa relazioni quando non esiste una chiara corrispondenza fra gli elementi dei frame da richiedere una modellazione tramite la relazione *Inheritance*. Per esempio, il frame *Judgment_communication* usa sia *Judgment* sia *Statement*, ma non eredita da loro nulla.
- *See_also* : collega frame che all'apparenza possono sembrare simili ma che necessitano comunque una distinzione.

I vantaggi derivanti dall'utilizzo di FrameNet sono molteplici. Avendo frame che non sono legati in nessun modo alle parti del discorso rende l'approccio di costruzione di questa risorsa lessicale più orientata ad aspetti cognitivi che linguistici. Soprattutto le relazioni definite fra frame diversi aprono la strada a diverse e nuove forme di ragionamento. Infine come punto di debolezza è opportuno notare come, mancando in FrameNet un controllo formale, non vi siano garanzie sulla correttezza a livello di rete della conoscenza rappresentata al suo interno.

Differenze fra WordNet e FrameNet

Se a WordNet è ormai riconosciuto il primato come risorsa più diffusa nella comunità della linguistica computazionale, FrameNet sta lentamente attirando a sé nuovi interessi. Come già accennato prima, il grande vantaggio di WordNet è la sua ampia copertura, mentre a FrameNet si può invece riconoscere il merito di aver costruito una più articolata e ricca analisi semantica del linguaggio che ha trovato la sua realizzazione nella rete di frame offerta. Soprattutto l'analisi svolta da FrameNet non si perde dietro i mille rivoli che una documentazione delle sottili sfumature del significato delle parole potrebbe richiedere. Invece questa risorsa cerca il più possibile di generalizzare i significati dei termini in oggetti che, seppur astratti, rimangono ben strutturati attraverso la definizione di frame. Ovviamente questo processo d'analisi richiede molte risorse, soprattutto in termini di tempo. La conseguenza di ciò è che la copertura, per quanto recentemente estesa con la versione 1.6, è comunque limitata.

Un loro uso combinato sarebbe auspicabile anche in virtù del fatto che esistano molti punti di similarità nell'informazione in esse contenuta. Ciò sarebbe una piacevole soluzione poiché i punti di forza di entrambe le risorse sono per loro natura ortogonali. Per WordNet le relazioni fra i synset sono gli aspetti più informativi, mentre FrameNet fornisce una accurata specializzazione delle relazioni presenti negli eventi. Sfortunatamente i tentativi finora proposti non hanno prodotto grandi risultati: una delle più accreditate proposte, il progetto *SemLink*[2], ha infatti proposto finora un allineamento delle informazioni per le sole forme verbali.

[2] https://verbs.colorado.edu/semlink/

4.2.3 ConceptNet

ConceptNet è una rete semantica che ha come obiettivo la descrizione della conoscenza tipicamente attribuita al "senso comune". Per descrivere tale tipo di conoscenza ConceptNet usa principalmente parole ed espressioni nel linguaggio naturale. Frasi e parole sono collegati fra di loro da predicati generici, come ad esempio *IsA* o *UsedFor*, riconducibili al nostro *comune senso* di ragionare.

Le scelte modellistiche adottate rendono ConceptNet particolarmente adatto per applicazioni di comprensione del linguaggio naturale. Come descritto sopra, i concetti sono connessi e definiti da parole e frasi estratte da testi reali; tali espressioni includono non solo definizioni e relazioni lessicali, ma anche relazioni comunemente riferite al nostro senso comune. E' però importante sottolineare come la maggior parte dei termini contenuti in ConceptNet non siano entità denominate individualmente, ma invece parole comuni per lo più estratte da *Wiktionary*[3].

Altro aspetto su cui porre attenzione riguarda la possibilità di trovare legami fra concetti esplicitati da relazioni "informali". Se tipicamente tali relazioni sono tralasciate nella progettazione di una risorsa lessicale, qui invece ricoprono un ruolo equivalente ad altri tipi di relazione poiché ad esse è attribuita parte della conoscenza quotidiana. Per esempio, il termine *jazz* è caratterizzato in ConceptNet non solo dalle proprietà che lo definiscono, come *IsA(jazz,popular music genre)*, ma anche da un insieme di proposizioni incidentali come *UsedFor(saxophone, jazz)* o *AtLocation(jazz, new orleans)*. Un piccolo esempio di relazione per il concetto *car* è mostrato in figura 4.1.

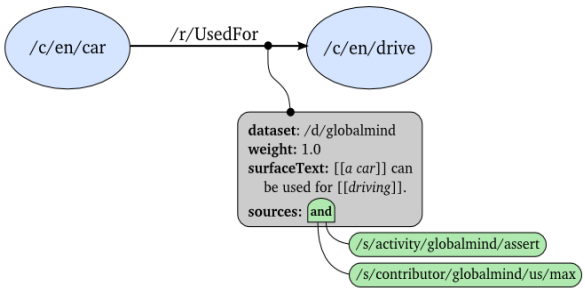


Figura 4.1. Esempio di relazione fra concetti in ConceptNet.

Come si vede in figura 4.1, i concetti sono espressi tramite parole e sono accompagnati da asserzioni sul modo in cui essi sono relazionati. Queste asserzioni possono essere estratta da varie fonti e tali fonti sono poi annotate e utilizzate come loro giustificazione.

[3] https://www.wiktionary.org/

La struttura dell'informazione contenuta in ConceptNet può essere rappresentata con un iper-grafo nel quale i nodi rappresentano i concetti e gli archi le asserzioni che li legano. Tali asserzioni sono etichettate con un insieme di relazioni, la cui lista completa è fornita in appendice. I concetti invece sono costruiti a partire dalle versioni normalizzate delle parole, rimuovendo le parole più frequenti e trasformando nella loro forma multi-parola i termini espressi seguendo la tecnica *CamelCase*.

Ogni concetti è rappresentati da un *URI*, come "*/c/en/run/n/basement*", che codifica il tipo(/c concetto, /r relazione, /a asserzione), la lingua(/en), il testo normalizzato(/run), la parte del discorso(/n), e opzionalmente un elemento per risolvere la possibile ambiguità(/baseball). Infine ogni URI di un asserzione contiene tutte le informazioni necessarie per la ricostruzione dell'asserzione. Nell'esempio dell'asserzione "*jazz is a kind of music*" ha come URI:

$$/a/[/r/IsA/, /c/en/jazz/, /c/en/music/]$$

Tuttavia è importante sottolineare come tale modo di rappresentare la conoscenza di senso comune non sia esente da insidie. Il modo quotidiano di parlare e di ragionare è caratterizzato da una notevole flessibilità e genericità. Tali peculiarità minano, in principio, l'obiettivo di ConceptNet. Come discusso nel capitolo precedente durante la discussione sulla semantica lessicale, non importa quanti dati ConceptNet riuscirà a rappresentare, il suo contenuto rimarrà comunque deficitario di qualche aspetto riferibile al senso comune. Ciononostante, accantonando quindi l'idea di una completa e profonda comprensione del linguaggio, ConceptNet risulta essere un valido supporto per le applicazioni che utilizzano il linguaggio naturale.

4.2.4 Altre risorse

Propositional Bank

Il progetto *Propositional Bank* (PropBank) nasce con l'obiettivo di aggiungere un livello di annotazione semantico alla collezione di testi annotati sintatticamente *Penn Treebank*. Il livello aggiunto fornisce informazioni sui ruoli semantici e una consistente descrizione delle strutture predicato-argomento di diverse realizzazioni sintattiche.

Tuttavia PropBank, non definendo un insieme universale di ruoli semantici, costruisce il suo inventario con ruoli che sono dipendenti dai singoli verbi. Gli argomenti di ciascun verbo sono numerati secondo la metodologia proposta da Dowty. Secondo questo approccio arg_0 definisce l'agente prototipale, arg_1 l'oggetto o il tema, e così via. I limiti di tale approccio sono ben noti, e per verbi con un alto grado di *valenza* non esiste una consistente generalizzazione.

La struttura PropBank è composta da un elemento per ogni verbo presente in Penn Treebank. Nel caso di verbi con più sensi, questi elementi sono collegati ad un insieme di frame (diversi da quelli in FrameNet) che enumerano

i ruoli da attribuire alle realizzazioni nel testo. Ogni ruolo è descritto da un piccolo testo che preclude i tipici approcci di ragionamento automatico. Infine l'unica interessante informazione ricavabile è legata al collegamento fra verbi e frame, ma questa informazione è anche alla base del progetto *VerbNet*.

VerbNet

Basandosi sulla classificazione dei verbi proposta da Levin, il progetto *VerbNet* si propone di offrire per classi di verbi un collegamento fra sintassi e semantica. Ogni classe contiene un insieme di verbi caratterizzati da una struttura di argomenti simili. Tali argomenti possono essere percepiti come i classici ruoli semantici, ma, a differenza di altre risorse, VerbNet fornisce dei vincoli più stringenti sugli attributi.

I vincoli sono organizzati in una piccola struttura gerarchica, circa 40 elementi, che caratterizzano maggiormente i ruoli. Ad esempio, per la classe denominata"$hit-18.1$" i ruoli associati sono:

$$\begin{aligned} roleSet_{hit-18.1} = \{\ & Agent[+intentionalcontrol], \\ & Patient[+concrete], \\ & Instrument[+concrete]\} \end{aligned}$$

Per ogni classe verbale sono descritti diversi schemi sintattici. Questi descrivono le possibili realizzazioni che i verbi possono avere all'interno di un'espressione linguistica. Ogni schema è formato da un verbo, un insieme di ruoli tematici ordinati secondo le preferenze imposte dal verbo in questione, e opzionalmente da altri elementi lessicali che potrebbero essere necessari per alcuni costrutti sintattici o loro alterazioni.

La semantica associata allo schema può essere interpretata come una congiunzione di predicati. Nell'esempio precedente gli schemi definiti sono della forma:

$$\begin{aligned} synFram_{hit-18.1} = \{\ & Agent\ V\ Patient, \\ & Agent\ V\ at\ Patient, \\ & Agent\ V\ Patient[+plural]\} \end{aligned}$$

mentre la semantica associata a "$Agent\ V\ Patient$" è espressa come:

$$\begin{aligned} sem_{Agent\_V\_Patient} = & \ cause(Agent, Event), \\ & manner(during(Event), \\ & directed\_motion, Agent), \\ & contact(end(Event), Agent, Patient), etc. \end{aligned}$$

Allo stato attuale, sia i predicati verbali che le restrizioni ad essi associati sono rappresentati con semplici stringhe testuali. Manca quindi una concreta formalizzazione assiomatica, che restringe di conseguenza l'applicabilità della risorsa. Come accade per PropBank, l'utilizzo di VerbNet è infatti limitato all'utilizzo degli schemi sintattici che legano gli argomenti dei verbi ai ruoli semantici. Tuttavia a differenza di PropBank, la generalizzazione offerta

attraverso le classi verbali può rivelarsi utile all'interno di un sistema come ontologie dove le entità formali possono trovare una loro caratterizzazione nel linguaggio naturale.

4.3 Ontologie

> "An ontology is an explicit specification of a conceptualization."

Benché questa definizione, proposta da Gruber, sia comunemente la più utilizzata per descrivere un'ontologia, la seguente definizione proposta successivamente da Guarino sottolinea alcuni importanti aspetti della definizione ontologica:

> "An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, i.e. its ontological commitment to a particular conceptualization of the world. The intended models of a logical language using such a vocabulary are constrained by its ontological commitment. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating these intended models."

Nell'elaborare questa definizione, Guarino impone una prospettiva formale intensionale allo sviluppo di un'ontologia. Seguendo questa prospettiva, la concettualizzazione di un aspetto del mondo è indipendente dal linguaggio, mentre un'ontologia, in quanto artefatto cognitivo, è necessariamente dipendente dal linguaggio. In altre parole, la concettualizzazione, qualunque essa sia, è indissolubilmente "vincolata"[4] dalla scelta del linguaggio.

Lo scopo primario delle ontologie è permettere inferenza logica sul contenuto che esse rappresentano. La definizione, riportata sopra, definisce un'ontologia come un insieme di assiomi che formalmente modellano il significato degli elementi di un vocabolario. Tale conoscenza è costruita a partire da un insieme di classi, relazioni ed individui.

Tradizionalmente le ontologie sono impegnate a rappresentare una particolare vista di un determinato dominio d'interesse. Il loro scopo primario è la rimozione dalla modellazione del dominio di ogni fonte di ambiguità. Cioè, ciò che è definito all'interno di un'ontologia è definito in modo univoco con un unico senso associato.

Un tale approccio è ortogonale a quello perseguito dalle risorse lessicali, che invece sono rivolte ad esplicitare i diversi sensi e ruoli che un determinato

[4] Vincolare è forse usato impropriamente qui. Il punto della questione è qui il legame, forse necessario, che sussiste fra concettualizzazione e linguaggio. Trattare questo punto non è semplice, e va ben oltre i modesti obiettivi alla base di questo lavoro.

termine assume in diversi contesti. Tuttavia, posto che le ontologie modellino informazioni specifiche di domino, avere una collezione di queste ontologie, ognuna indipendente dall'altra, non si traduce in un immediato beneficio poiché allineare le informazioni in esse contenute non è sempre possibile. Risolvere questo problema è lo scopo alla base delle ontologie fondamentali.

4.3.1 Ontologie fondamentali

Lo scopo principale dietro lo sviluppo delle ontologie fondamentali è permettere un'interoperabilità semantica fra differenti ontologie specifiche di dominio. Un'ontologia fondamentale, anche detta *upper* o *top-level ontology*, introduce concetti generali che sono supposti essere condivisi fra differenti domini. Una volta adottata un'ontologia di alto livello, le ontologie di domino, allineate con quest'ontologia fondamentale, possono operare come un'unica risorsa permettendo un ragionamento condiviso nonostante le concettualizzazioni iniziali possano essere in principio incompatibili.

L'unico requisito imposto riguarda l'universalità dei concetti modellati nell'ontologia fondamentale; essa deve garantire per ogni concetto del dominio un'adeguata categoria. Tuttavia è importante sottolineare come le scelte ontologiche operate dalle ontologie fondamentali, anche se in modo indiretto, condizioneranno il qualche misura la conoscenza delle ontologie di dominio. Ciò avviene perché attraverso le categorie di alto livello si proiettano sull'ontologia di dominio i principi di modellazione adottati nell'ontologia fondamentale.

4.3.2 Dolce

Dolce è l'acronimo di *Descriptive Ontology for Linguistic and Cognitive Engineering*. L'ambizioso traguardo dietro la progettazione di Dolce è il tentativo di catturare le categorie fondamentali del linguaggio naturale e del comune modo di ragionare delle persone.

Dolce non tenta di catturare la natura *ultima* delle entità presenti nel mondo. Invece si propone di descrivere entità cognitivamente percepite che risultano cruciali per il comune modo di ragionare. Le categorie di Dolce sono considerate delle nozioni descrittive esplicitanti legami ed entità già note.

L'inclinazione verso il cognitivismo per catturare le categorie legate al linguaggio naturale e al senso comune permette a Dolce di cogliere le diverse rappresentazioni della realtà. Questa realtà è il frutto della percezione umana, dello sfondo culturale e alle convenzioni sociali. Dolce può essere quindi definita come un'ontologia di *particolari*, nella misura in cui il dominio del discorso è ristretto ad essi. Dolce infatti riserva il termine *universale* alle sole relazioni e proprietà sulle quali poter organizzare e caratterizzare i particolari.

Un altro aspetto interessante di DOLCE è il suo approccio moltiplicativo, secondo il quale entità distinte possono essere co-localizzate nella stessa regione spazio-temporale. Ciò permette di modellare situazioni molto comuni, come ad esempio la descrizione di un *piatto di ceramica*. Quest'entità può quindi

essere scomposta in diverse sotto-entità che in momenti diversi possiedono delle proprietà differenti(magari incompatibili fra di loro), ma che raggiungo comunque un loro equilibrio nel loro stato finale.

La fondamentale distinzione che Dolce opera sulle sue entità è la loro suddivisione in *endurant*(continuante) e *perdurant*(occorrente). La distinzione è qui posta lungo una dimensione temporale. Gli endurant sono interamente[5] presenti in ogni istante della loro presenza. Mentre i perdurant possono essere presenti nel tempo solo attraverso un successivo accumulo di diverse loro parti costituenti, ovvero in ogni istante della loro presenza sono solo parzialmente presenti. Un esempio di quest'ultima tipologia abbastanza intuitivo è l'evento della lettura di un libro.

Gli endurant possono cambiare nel tempo e partecipare nei perdurant. Altro elemento di distinzione fra endurant e perdurant è legata alla possibilità dei primi di potersi modificare. Infatti gli endurant pur sperimentando cambiamenti non perdono la loro identità, mentre i perdurant, poiché occorrono nel tempo, sono condannati dalla mutazione a diventare una nuova entità. In questo contesto la relazione che lega endurant e perdurant è basata sul concetto di partecipazione.

Un altro elemento distintivo di Dolce riguarda la trattazione delle qualità e dei valori, denominati *qualia*, associate alle sotto-categorie di endurant e perdurant. Le qualità, pur essendo nella prospettiva di Dolce le principali entità che è possibile misurare e percepire (forma, colore, dimensione, suono, odore, ecc.), non sono considerate come degli universali poiché distinte dal concetto di proprietà.

In Dolce esiste una chiara distinzione fra una *qualità individuale*, un *quale*, e *qualità spazialmente definite*. Le qualità individuali, come il colore di un libro, "appartengono" agli individui; ovvero non importa quanto simili ai sensi due libri rossi possano apparire, le loro qualità individuali sono differenti(benché possono avere lo stesso valore di qualità). Essendo quindi le qualità legate alle entità, ne deriva che la mancanza di queste ultime implichi l'immediata cessazione d'esistenza delle qualità ad esse riferite.

Diversamente dalle qualità individuali, le *qualia* sono indipendenti dall'entità. L'esempio più semplice di un *quale* è un'entità rappresentante uno specifico colore come il rosso. Intuitivamente, queste entità sono ottenute attraverso un processo di astrazione temporale e materiale delle qualità individuali. In questo contesto allora le qualia permetto di esprimere similarità perfette e oggettive fra due oggetti. Le qualia sono organizzate in modo differente all'interno di diversi *spazi qualitativi*. Motivati dalle similarità fra gli oggetti o fra alcuni loro aspetti, la presenza di diversi spazi per la stessa qualità garantisce comunque la possibilità di imporre differenti strutture ai qualia, offrendo la possibilità di differenziare quantitativamente e qualitativamente i diversi gradi di similarità.

[5] Ovvero insieme a tutte le loro parti costituenti.

Spazio e tempo, nei rispettivi spazi concettuali, sono anche essi considerati come qualità. Ma all'interno di Dolce essi ricoprono un ruolo particolare in quanto tutte le qualità derivano le qualità spazio-temporali dei partecipanti ad un'entità. Ma è opportuno sottolineare come nessuna relazione di appartenenza sia definibile per le qualità nell'ontologia.

Infine l'ultima categoria, che Dolce offre per caratterizzare le entità, è quella *astratta*. Le entità classificate come astratte non hanno qualità spazio-temporali e non sono considerate come delle qualità. Un esempio di tale classe astratta sono gli stessi spazi qualitativi.

La figura 4.2 mostra l'ulteriore suddivisione in sotto-categorie. Gli endurant sono divisi in *Physical* e *Non-Physical*, Mentre i primi sono caratterizzati da qualità spaziali dirette, scomposte poi in *Amounts of matter, Objects* e *Features* a seconda del loro tipo, gli altri invece sono dapprima categorizzati da *Mental* e *Social Objects*, e successivamente da *Agentive* e *Non-Agentive* a seconda della loro intenzionalità.

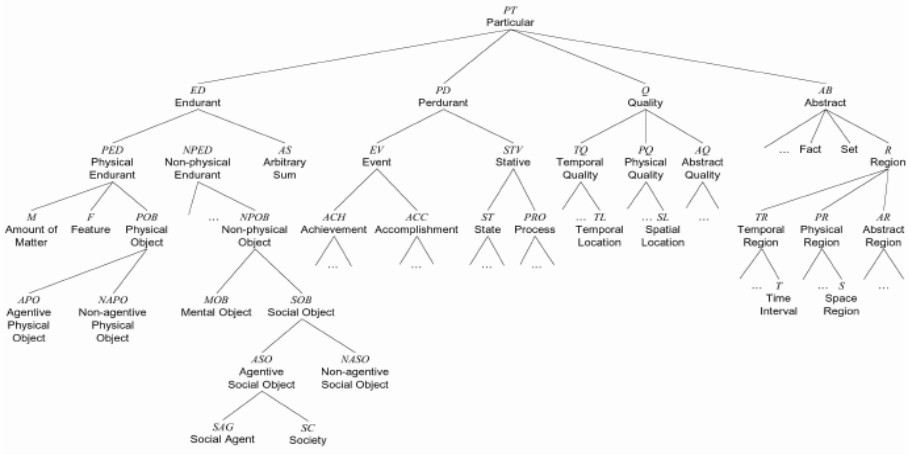


Figura 4.2. Tassonomia delle categorie di Dolce.

Nei perdurant invece rientra tutto ciò che può essere definito come evento, processo, attività o stato. Questi sono divisi seguendo il principio di *homeomericity*, cioè esiste un'occorrenza se e solo se tutte le sue parti temporali possono essere descritte nello stesso modo per tutta la sua durata. Nel caso ciò non fosse possibile, allora essi sono descritti come *cumulativity*, ovvero l'occorrenza è il risultato della somma di più istanze.

Col tempo si è però visto come l'adozione di Dolce nella sua interezza poteva tradursi in un maggiore ostacolo per la sua diffusione. Per questo motivo è stata creata una sua versioni *Lite* rivolta a semplificare la struttura completa di DOLCE. Questa versione semplificata è stata successivamente utilizzata anche nel progetto rivolto alla formalizzazione di WordNet: OntoWordNet.

4.3.3 Sumo

La *Suggested Upper Merged Ontology* è il risultato della fusione di contenuti provenienti da diverse ontologie all'interno di un'unica struttura. SUMO non propone nessun nuovo principio di modellazione della conoscenza, ma si limita semplicemente all'integrazione di risorse già esistenti. Fra queste risorse vi sono l'ontologia fondamentale proposta da Sowa, quella di Russell e Norvig, gli assiomi temporali di Allen, l'ontologia dei confini di Smith e Dolce.

Attualmente Sumo è diviso in 11 sezioni con interdipendenze attentamente specificate e rappresentate in figura 4.3. La prima sezione, denominata ontologia strutturale, contiene le definizioni per le relazioni che servono alla costruzione di un sistema di ontologie. La seconda sezione, denominata *Base Ontology*, ha lo scopo di definire i concetti atomici, come ad esempio la nozione di entità astratta o la differenza fra oggetto e processo.

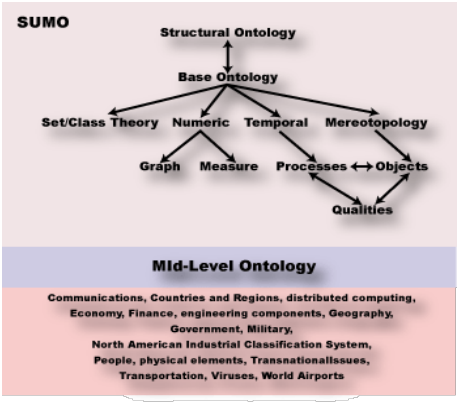


Figura 4.3. Architettura ontologica di Sumo.

Nelle ontologie di base ogni sezione è incentrata su uno specifico tema. Come mostrato dalla figura 4.3, sono presenti nozione astratte come quelle riferite alla teoria degli insiemi e dei grafi, alle funzioni aritmetiche di base, relazioni temporali, o assiomatizzazioni di relazioni parte-tutto, e così via. Le restanti sezioni forniscono invece gerarchie e tipi di processo, di oggetti e di attributi. Infine vi è un livello intermedio che funge da interfaccia con un insieme di ontologie specifiche di dominio come *comunicazione*, *economia*, etc.

Le categorie principali che Sumo utilizza per classificare le entità sono *Physical* e *Abstract*. Se la prima è impiegata per caratterizzare la proprietà spazio-temporale di una qualsiasi entità fisica(che può essere ulteriormente specificata poi nelle sotto-categorie *Object* e *Process*), nella seconda rientrano

tutte le entità escluse dalla precedente categoria. Una lista dei primi quattro livelli è riportata di seguito.

- *Physical*
  - *Object*
    - *Collection*
    - *SelfConnectedObject*
      - *ContinousObject*
      - *CorpuscularObject*
  - *Process*
- *Abstract*
  - *Set Class*
    - *Relation*
  - *Proposition*
  - *Attribute*
  - *Quantity*
    - *Number*
    - *PhysicalQuantity*

La gerarchia di classi e relazioni sono definite nel linguaggio *SUO-KIF32* che è caratterizzato da una semantica dichiarativa leggermente superiore al primo ordine. La sua ricca tassonomia, ha permesso a Sumo di essere impiegato in applicazioni per la comprensione del testo, soprattutto dopo l'allineamento con WordNet.

4.3.4 Yago

Yago è una delle più grandi ontologie pubbliche costruite con l'ausilio di tecniche di estrazione dell'informazione utilizzando le pagine di Wikipedia come fonte di conoscenza principale. Tuttavia Yago è stato successivamente esteso così da comprendere al suo interno conoscenza proveniente anche da altre risorse pre-selezionate.

Come accade per altre basi di conoscenza(TrueKnowledge, Freebase, il Knowledge Graph di Google, etc.) che estraggono informazioni da risorse web, Yago memorizza la sua conoscenza in *triple*. Una tripla è una asserzione *soggetto-predicato-oggetto*, un esempio per l'informazione che "Barack Obama was born in Honolulu" è rappresentato dalla tupla associabile $\langle Barack\_Obama, born\_in, Honolulu \rangle$.

A differenza di altre risorse la precisione dell'informazione contenuta è molto alta. Yago infatti estrae informazione utilizzando solo un particolare insieme di relazioni manualmente selezionato. Tali relazioni sono scelte fra quelle messe a disposizione dalle diverse risorse utilizzate. Da ciò deriva un livello di precisione molto alto che è stato manualmente valutato mostrando come fosse stato possibile raggiungere il 95% di correttezza nelle triple di Yago. Ovviamente raggiungere un tale livelli di correttezza non è in generale semplice, ma nel caso di Yago ciò è stato possibile grazie all'utilizzo di

un numero ristretto di risorse attentamente selezionate: *Wikipedia, WordNet, Geonames, Universal WordNet*, e *WordNet Domains* solo per citarne alcune.

Nel caso di Wikipedia, ad esempio, il titolo delle pagine contribuisce alla definizione delle entità di Yago. Le categorie degli articoli di Wikipedia sono utilizzate per derivare i tipi delle entità. Le parti speciali delle pagine sono elaborate per estrarre fatti sulle entità. Mentre la tassonomia di WordNet fornisce la struttura di base per organizzare l'informazione. Ogni fatto è controllato rispetto alla definizione di tipo delle relazioni interessate e ai vincoli di funzionalità. Ad ogni entità, infatti, deve essere necessariamente assegnato un tipo il più possibile preciso poiché il sistema vincola marcatamente l'inter-compatibilità fra i tipi.

Con la versione 2, Yago subì una completa re-ingegnerizzazione della sua architettura perseguendo un approccio più modulare. La nuova architettura è basata su temi, o collezione di fatti. Sono presenti piccoli moduli software, denominati estrattori, che generano nuovi temi a partire da quelli precedenti. Esempi di tali estrattori possono essere i verificatori di tipo, eliminatori di duplicati o semplici aggregatori d'informazione. La loro azione combinata garantisce che tutti i fatti memorizzati nel livello più alto dell'architettura siano consistenti e unici. L'insieme dei temi nel livello finale costituisce l'ontologia Yago.

4.4 BabelNet

BabelNet è il più ampio dizionario enciclopedico multilingua attualmente disponibile. Tutti i termini in esso contenuti partecipano alla definizione di una ampia rete semantica che fornisce utili informazioni in applicazioni riguardanti il linguaggio naturale.

BabelNet nasce primariamente nel tentativo di integrare Wikipedia, che rappresenta attualmente il più popolare lessico per la lingua inglese, e WordNet. Col tempo lo sviluppo del dizionario ha permesso di estendere il processo d'integrazione anche ad altre risorse, il cui elenco è riportato in tabella 4.2. L'innovazione proposta da BabelNet risiede principalmente nell'integrazione automatica di tutte le risorse sopra elencate. Il risultato di tale integrazione è un enorme quantità di concetti ed entità interconnessi attraverso relazioni semantiche.

Essendo BabelNet rivolto verso una completa copertura lessicografica ed enciclopedica, può tranquillamente essere visto come un valido complemento a risorse come DBPedia, YAGO o WikiNet. Combinato con YAGO, ad esempio, BabelNet si riesce, attraverso l'utilizzo di uno specifico algoritmo per la risoluzione del senso di una parola, a fornire più realizzazioni linguistiche(anche su diverse lingue) per lo stesso concetto.

La conoscenza all'interno di BabelNet è strutturata come un grafo diretto $G = (V, E)$, i cui nodi rappresentano concetti (tipo *movie*), o entità(come *blues brothers*), mentre i sui archi $E \subseteq V \times R \times V$ connettono coppie di nodi

Tabella 4.2. Risorse impiegate da BabelNet

Risorsa	Descrizione
Open Multilingual WordNet	una collezione di Wordnet in diverse lingue
OmegaWiki	dizionario multi-lingua costruito in modo collaborativo
Wiktionary	dizionario multi-lingua senza contenuti
Wikidata	base di conoscenza rivolta alla strutturazione della conoscenza contenuta in Wikipedia
Wikiquote	sorgente di citazioni in più linguaggi
VerbNet	descritta sopra
Microsoft Terminology	collezioni di termini utilizzati per lo sviluppo localizzato di applicazioni
GeoNames	database geografico
WoNeF	una versione migliorata ed estesa di una traduzione automatica di WordNet per il francese
ItalWordNet	database lessicale per la lingua italiana
ImageNet	un database di foto organizzate secondo la gerarchia di WordNet
FrameNet	descritta sopra

attraverso delle relazione rappresentanti proprietà o legami semantici. Ogni relazione ha associato un'etichetta che corrisponde ad elementi dell'insieme delle relazioni semantiche $R = \{is\_a, part\_of, \ldots, \epsilon\}$, dove ϵ rappresenta una relazione sconosciuta. Ogni nodo è poi corredato dalle sue diverse realizzazioni in diverse lingue. L'unione delle risorse sotto un unico concetto rappresenta il BabelNet *synset*.

L'approccio adottato da BabelNet si basa sull'assunzione che esiste, in una qualche misura, una sovrapposizione dell'informazione fra le diverse risorse. Restringendo la discussione, per questioni di semplicità, alle sole risorse WordNet e Wikipedia si descrive il processo di estrazione dell'informazione operato da BabelNet. Concetti e relazioni sono estratti automaticamente seguendo un approccio suddiviso in tre fasi:

1. WordNet e Wikipedia sono combinati insieme dall'acquisizione automatica di un allineamento fra i sensi di WordNet e le pagine di Wikipedia. Ciò elimina la duplicazione dei concetti e permette di estendere il lessico finale.
2. Per i synset costruiti nella fase precedente si estraggono le relative espressioni in altre lingue sfruttando le versioni multilingue delle pagine di Wikipedia, e i sistemi di traduzione automatici.
3. Le relazioni fra i synset di BabelNet sono costruite a partire da quelle presenti in WordNet, e dai riferimenti presenti nelle pagine di Wikipedia. La misura di correlazione fra i synset utilizzata da BabelNet è basata sul coefficiente di Dice.

Un esempio del risultato di tale costruzione è mostrato in figura 4.4.

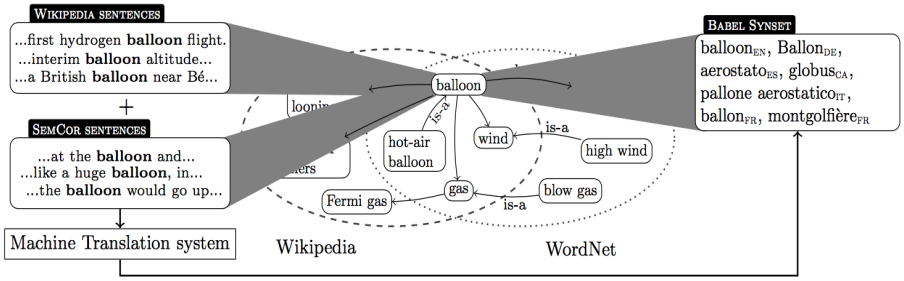


Figura 4.4. Esempio di nodi appartenenti alla rete semantica di BabelNet. Alcuni archi sono etichettati poiché derivano da WordNet, mentre gli altri, senza descrizione, derivano da Wikipedia.

Riassumendo, tutti i sensi estratti da WordNet diventano concetti e tutte le relazioni fra synset di WordNet rimangono tali anche in BabelNet. Da Wikipedia invece sono estratti tutti i riferimenti enciclopedici che sono comunque trattati come concetti, mentre i loro collegamenti ipertestuali diventano relazioni semantiche non specificate.

Benché non etichettate, tutte le relazioni provenienti da Wikipedia sono comunque trattate come relazioni semantiche alla pari di quelle estratte da WordNet. Ciò accade perché, come si può vedere dalla tabella4.5, Wikipedia rappresenta la maggiore fonte di relazioni presente in BabelNet.

Language	WordNet	Wikipedia	WordNet glosses	Wikidata	WiBitaxonomy	Wikipedia Infoboxes
EN English	364569	71281909	616447	20743527	6337743	13089487
DE German	0	24860448	0	1024162	0	474854
JA Japanese	0	23792810	0	246626	0	200372
FR French	0	20843237	0	909882	0	513927
RU Russian	0	15346803	0	616403	0	273092
IT Italian	0	15190790	0	432081	0	264440

Figura 4.5. Numero di relazioni semantico lessicali per i primi sei linguaggi presenti in BabelNet.

In aggiunta all'insieme di relazioni, BabelNet fornisce per ogni synset uno o più descrizioni, dette *gloss*, che esemplificano il significato del synset. Ricavare quest'informazione da WordNet è immediato poiché WordNet fornisce a sua

volta un gloss per ognuno dei suoi synset. Per Wikipedia invece, la definizione testuale è estratta dalla prima frase presente in ogni sua pagina.

Statistiche

Le statistiche mostrate in figura 4.6 si riferiscono alla versione 3.7 di BabelNet.

Number of languages:	271
Total number of Babel synsets:	13,801,844
Total number of Babel senses:	745,859,932
Total number of concepts:	6,066,396
Total number of Named Entities:	7,735,448
Total number of lexico-semantic relations:	380,239,084
Total number of glosses (textual definitions):	40,709,194
Total number of images:	10,767,833
Total number of Babel synsets with at least one domain:	2,675,385
Total number of compounds:	743,296
Total number of other forms:	6,393,568
Total number of Babel synsets with at least one picture:	2,948,668
Total number of RDF triples:	1,971,744,856

Figura 4.6. Statistiche per BabelNet v 3.7 .

4.4.1 WiBiTaxonomy

A partire dalla versione 3.0 BabelNet è completamente tassonomizzata. Ciò è stato possibile dall'integrazione con il progetto *Wikipedia Bitaxonomy* che è rivolto ad estrarre ed allineare automaticamente due tassonomie : una per le pagine di Wikipedia, ed una per le sue categorie.

Da un'estensiva comparazione con MENTA, DBpedia, YAGO, WikiTaxonomy e WikiNet su due database contenenti 1000 pagine e categorie, è stato possibile verificare l'alta qualità(in termini di copertura e specificità) delle tassonomie prodotte che si traduce in un alto indice di *precisione* e *recupero*.

L'approccio per costruire le bi-tassonomie per la versione inglese di Wikipedia si struttura su tre fasi principali:

1. Usando un insieme di *linker* la tassonomia delle pagine è popolata. Le definizioni testuali sono elaborate e sono estratte le iperonimie, che sono successivamente risolte semanticamente in relazione alla pagina sotto esame.
2. Usando un particolare algoritmo la tassonomia delle categoria è popolata mentre la tassonomia delle pagine è iterativamente arricchita.

Per indurre la tassonomia sulle categorie di Wikipedia si sfruttano le iperonimie della tassonomia delle pagine e i loro collegamenti verso le categorie. In questo modo si induce una tassonomia sulle categorie in modo iterativo. Ad ogni iterazione, i collegamenti nella tassonomie delle pagine sono usati per identificare iperonimie fra le categorie, e successivamente questi iperonimi delle categorie sono utilizzati per arricchire la tassonomia delle pagine.

3. Le categorie tassonomiche sono poi sottoposte ad una serie di raffinamenti strutturali. Un insieme di euristiche strutturali sono applicate per risolvere problemi di ereditarietà che possono affliggere le categorie.

Il risultato di quest'approccio è una bi-tassonomia di milioni di pagine e centinaia di migliaia di categorie.

4.5 Sommario e ulteriori riferimenti

Una buona introduzione alle risorse lessicali è il lavoro di Miller Fellbaum[106], mentre una discussione approfondita sul tipo di relazione da impiegare può essere trovata in [146]. WordNet è stato introdotto in [105] e successivamente in [104, 42]. Informazioni sulla rete semantica FrameNet possono essere trovate in[130], mentre l'introduzione della semantica dei frame è riconducibile al lavoro di Fillmore in [44].

ConceptNet nasce come rappresentazione della conoscenza raccolta nel progetto *Open Mind Common Sense* [138], che era basato su un'interfaccia web per la raccolta di nuove affermazioni o sulla loro convalida. La prima versione di ConceptNet è descritta nel lavoro [91], mentre la sua versione più recente, ConceptNet 5.5, è invece discussa in [139]. Infine una discussione delle sue possibile applicazioni è fornita in [143].

Il progetto Propositional Bank è descritto nel lavoro [117], mentre il modello utilizzato per la strutturazione dei ruoli semantici basato sull'approccio proposto da Dowty è presentato in [35]. Il progetto VerbNet è invece fondato sulla classificazione verbale suggerita da Levin in [86], e la sua descrizione è presentata in [78].

La discussione sulle ontologia non può non partire da lavoro di Gruber [57]. Tuttavia i lavori di Guarino sul tema sono sicuramente degni di interessi e approfondimento; fra questi è possibile indicare il lavoro [58] all'interno del quale l'autore definisce in dettaglio il ruolo delle ontologie sfruttando la loro applicazioni in determinati campi del recupero dell'informazione. I fondamenti ontologici dietro la costruzione di Dolce possono essere approfonditi in [10], mentre un esempio di applicazione di Dolce sulla risorsa lessicale WordNet è discusso in [51]. Sumo è presentata nel lavoro di Niles e Pease [114], e fu successivamente allineata con WordNet in [115]. Questa risorsa ha trovato interessanti applicazioni nel contesto dei sistemi automatici di risposta come [59, 3]. Infine maggiori informazione su Yago possono essere trovate in [95].

L'ultima risorsa discussa in questo capitolo è BabelNet. Questo dizionario enciclopedico, discusso ampiamente in [113], è in continuo aggiornamento. Oltre all'integrazione con WiBiTaxonomy [46, 47], la versione attuale BabelNet 3.7, offre interessanti spunti per la rappresentazione semantica dei concetti [19, 67].

5

Il problema di generare una risposta ad una domanda

In letteratura il problema di rispondere ad una domanda posta in linguaggio naturale è comunemente scomposto in due principali sotto-problemi: comprendere il significato della domanda e recuperare l'informazione necessaria a generare la risposta. Per loro natura questi sotto-problemi sono spesso affrontati con approcci più pragmatici che teorici. Oggigiorno è infatti abbastanza comune imbattersi in prodotti basati semplicemente su collezioni di strumenti e tecniche per la generazione delle risposte, anziché su una teoria formale che modelli tutti gli aspetti del problema e definisca chiaramente l'output atteso.

Generalmente, rispondere a domande naturali coinvolge diversi strumenti forniti dalla linguistica computazionale e dal recupero automatico dell'informazione. Per l'analisi linguistica della domanda sono molto importanti i processi di segmentazione, di annotazione sintattica e di individuazione delle entità. Strumenti come classificatori, indicizzatori e algoritmi di apprendimento automatico rivestono invece un ruolo più importante nel recupero dell'informazione.

Nel seguito si descriveranno brevemente alcuni sistemi di generazione di risposta e saranno forniti alcuni cenni ai sistemi presenti in letteratura; successivamente si introdurranno le diverse tipologie di domanda per una loro possibile classificazione.

5.1 Tipologia di Ricerca

Il dominio di ricerca riferito ai sistemi per generare risposte a domande poste in linguaggio naturale, nel seguito denominati sistemi QA, comprende i risultati delle ricerche nei campi del recupero dell'informazione, dell'elaborazione del linguaggio naturale, e dell'estrazione della conoscenza.

Sebbene questi sistemi possono essere a prima vista assimilati ai motori di ricerca presenti sul web, una comparazione più approfondita mostra come, pur condividendo alcuni tratti comuni, gli obiettivi di entrambi i sistemi siano abbastanza differenti. Lo scopo dei motori di ricerca è infatti fornire una lista

di documenti, o pagine web, come risposta ad una richiesta fatta da un utente. Molto spesso la ricerca dei documenti si basa su parole-chiave e misure di popolarità, di frequenza di accessi, affidabilità, e così via. Discorso simile vale per i sistemi di recupero dell'informazione. Questi sistemi infatti non generano risposte ma, come accade per i motori di ricerca, forniscono all'utente una lista ordinata di documenti all'interno dei quali è possibile trovare la risposta alla domanda posta. Tuttavia sarebbe auspicabile richiedere al sistema di ritornare non una lista di documenti ma poche rilevanti e concise frasi che siano "sufficienti" a rispondere esaustivamente alla richiesta. Affrontare questo problema è alla basa della ricerca per la definizione di un sistema QA.

In base al tipo d'informazione, o conoscenza, che i sistemi QA riescono a trattare è possibile suddividere quest'ultimi in tre distinte categorie. Sistemi che operano su domini aperti di conoscenza sono capaci di rispondere a tutti i tipi di domanda; ma, sfortunatamente, l'intrinseca complessità del problema limita la loro precisione e copertura. Sistemi, il cui dominio è chiuso, restringono la loro capacità di risposta a specifiche aree di conoscenza, utilizzando prevalentemente informazione molto strutturata spesso contenuta all'interno di basi di dati o ontologie. Infine esistono sistemi ibridi operanti su conoscenza specifica di dominio con capacità di ragionamento avanzate. In questo modo è possibile garantire elevati livelli di precisione in applicazioni reali.

In letteratura sono ormai presenti molti esempi di sistemi QA. Fra i primi sistemi proposti vi furono *baseball* e *lunar*. Entrambi i sistemi furono caratterizzati da una conoscenza altamente specifica e da un quantità d'informazione abbastanza esigua. Mentre il primo era capace di rispondere a domande su una particolare stagione del campionato americano di baseball, il secondo era rivolto a fornire informazioni sulle analisi chimiche condotte durante le missioni Apollo. Successivamente, l'avvento di nuove tecnologie per l'elaborazione del linguaggio permise di focalizzare l'attenzione anche su aspetti legati alla sintassi e alla semantica.

Il sistema START sviluppato Boris Katz, ad esempio, usò sia meccanismi basati su annotazioni del linguaggio naturale per selezionare le risposte da ritornare, sia strategie di decomposizione sintattica e semantica per poter operare su sorgenti multiple di informazione così da rimuovere ridondanza e similarità semantica delle richieste. Nei sistemi *lasso* e *falcon*, invece, le domande furono processate combinando le informazioni sintattiche e semantiche che le caratterizzavano. Durante tale analisi furono poi impiegate regole euristiche per estrarre parole-chiave utilizzate per l'identificazione della domanda.

Nel successivo sistema *QA-LaSIE* l'analisi del testo fu accompagnata da sistemi di recupero dell'informazione. In particolare l'approccio proposto eseguiva un'elaborazione semantica dei documenti, ritornati dai sistemi di recupero, così da selezionare il paragrafo più adatto per rispondere alla domanda.

Nonostante i molti anni di ricerca le prestazioni di questi sistemi, in termini di precisione e recupero, non furono molto soddisfacenti. Nel tentativo di migliorare questa situazione Zhang e Lee provarono ad applicare tecniche di apprendimento automatico come Support Vector Machine(SVM), Nearest

Neighbors(NN), Naive Bayes(NB), Decision Tree(DT) and Sparse Network of Winnows(SNoW). Alla fine della loro comparazione gli autori proposero una speciale funzione kernel che permetteva a SVM di sfruttare la struttura sintattica della domanda per classificare la domanda. Tuttavia solo recentemente il sistema Watson, proposto da IBM, ha dimostrato una notevole capacità interpretativa. Tale sistema è infatti capace di analizzare il linguaggio naturale, identificare le sorgenti d'interesse, trovare e generare ipotesi, trovare e valutare evidenze a supporto delle ipotesi e ritornare quella che riceve il valore di confidenza maggiore.

5.2 Classificazione delle domande

Sebbene riconoscere una domanda è un'operazione abbastanza comune per le persone, non è in generale possibile dare una definizione formale di cosa sia una domanda per un sistema QA. Ciononostante è possibile avanzare una prima distinzione sulla base delle richieste e delle metodologie utilizzate per ritornare una risposta come nel caso di "*Who is the new president of the U.S.A. ?*" e "*What is the square root of minus one?*". Su queste distinzioni è possibile assumere come la maggior parte degli approcci sia assimilabile ad una metodologia la cui natura è intrinsecamente *estrattiva*: ovvero gli approcci sono generalmente rivolti a trovare risposte che sono già note(presenti in qualche risorsa), invece di essere focalizzati sulla generazione di nuova conoscenza.

Molto spesso si utilizza il termine "*factoid questions*" per descrivere domande che si riferiscono alla richiesta di fatti o relazioni presenti in una base di conoscenza. Queste domande sono frequentemente caratterizzate dalla presenze delle parole "*Wh-*", come *what, where, when*, ma sono in generale escluse le domande, introdotte da "*Why*" e "*How*", che richiedono forme di ragionamento più complesse per costruire la risposta. In questo modo è possibile sfruttare le indicazioni che tali domande forniscono sulla classe di entità da ritornare desiderata. In altre parole, la preferenza per tali domande risiede infatti nella loro capacità di determinare implicitamente il tipo di risposta: espressioni come "*How far*" o "*When*", ad esempio, ricevono rispettivamente come classe di risposta una distanza e un riferimento temporale. All'interno di tali classi è poi possibile sollevare problematiche relative alla corretta granularità e alla scala di misura da utilizzare, ma generalmente questi aspetti ricoprono un ruolo secondario nel processo di risposta.

Tuttavia non tutte le domande di tipo "*Wh-*" offrono chiare indicazioni sulla natura del tipo di risposa da ritornare. Semplici domande come "*How did Socrates die?*" possono accettare come loro risposte semplici espressioni nominali tipo "*suicide*" o "*hemlock*", ma non escludono risposte articolate su più frasi che descrivono ulteriori aspetti contestuali. Domande come "*Where does sugar come from ?*" sono caratterizzate da una particolare struttura interna. Tali domande si possono infatti riferire sia al risultato finale del processo fisico che produce una certa *cosa*, sia alle materie prime utilizzate per la

sua realizzazione o al loro luogo di provenienza. Oppure, può anche accedere che il contesto della domanda richieda che la risposta alla domanda contenga una loro combinazione.

Altre tipologie di domande possono richiedere che la risposta desiderata sia l'insieme delle entità soddisfacenti una certa proprietà, o una collezione di fatti o caratteristiche riferite ad una certa entità. Le prime sono spesso riferite come "*list questions*", e la risposta loro associata definisce un insieme di fatti contenuti in una base di conoscenza o in un insieme di documenti. E' opportuno notare come a differenza delle domande che richiedono una singola risposta, qui, la risposta è composta da tutti i fatti che possono essere ritenuti in un certo modo corretti. Da ciò la necessità di assegnare non solo un punteggio ai singoli candidati ma anche una misura di confidenza con il contesto delineato dalla domanda. Nella seconda categoria invece ciò che è richiesto è una definizione, cioè una collezioni di fatti o proprietà necessari a descrivere una certa entità. Per esempio, la domanda "*Who is John Lennon*" potrebbe essere risposta con informazioni sulla sua carriera da cantautore, ma anche informazioni riguardanti il suo attivismo politico-sociale.

Infine un'altra classe di domande da considerare riguarda quelle il cui tema è focalizzato sull'esistenza di una certa relazione fra una coppia di entità. Per queste domande le risposte possono essere evidenze delle entità influenzate da una particolare entità, o della misura in cui le entità sono legate. Se nel campo dei sistemi per il recupero d'informazione tali risposte potevano limitarsi a semplici porzioni di testo contenenti entrambe le entità, nell'universo delle basi di conoscenza la risposta equivale alla selezione o costruzione di percorsi all'interno di reti semantiche per collegare le due entità .

Da queste semplici considerazioni è facile intuire come assegnare la corretta classe di risposta sia in generale più difficile della semplice classificazione sintattica degli elementi della domanda.

5.3 Terminologia

Come da prassi, per trattare più in dettaglio la questione riferita ai sistemi di risposta è opportuno introdurre una specifica terminologia. La discussione della sezione precedente evidenzia alcuni aspetti caratterizzanti la domanda e la risposta. Tuttavia tali aspetti, essendo stati principalmente trattati in maniera discorsiva, richiedono ora una definizione più precisa. Nel seguito si introducono quindi i termini per i diversi aspetti della domanda e della risposta che saranno utilizzati nel resto del capitolo.

Frase interrogativa

L'espressione di domanda è ciò che definisce cosa cercare. Nella sua forma più semplice questo costrutto è espresso dalle sole parole "*Wh-*", alle quali è attribuito il valore di soggetto principale della domanda. In generale, però, queste

frasi sono identificate dall'unione delle parole "*Wh-*" e dal nome(aggettivo o verbo) che esse modificano. Eccezioni a questa tipologia di costrutto sono le domande che richiedono una semplice affermazione o negazione.

Tipologia della domanda

Assegnare un tipo ad una domanda serve a distinguere le diverse strategie adottate per la costruzione della risposta. Nella tabella seguente sono riportate alcune categorie di domande discusse precedentemente. Si noti come, fra di loro i tipi, non sono mutualmente esclusivi e che enumerare tutte le possibili forme di interrogazione è una questione ancora irrisolta.

Tabella 5.1. Elenco delle tipologie di domande

Tipo	Esempio
FACTOID	What is the h-index of Dr. Rossi?
LIST	List the titles of books written by Prof. Bianchi.
DEFINITION	What is Artificial intelligence?
RELATIONSHIP	What is the connection between game theory and graph theory?
SUPERLATIVE	What is the most citated article?
YES–NO	Has Dr. Rossi any publication this year?
OPINION	Who is the greatest researcher in Computer Science?
CAUSE-EFFECT	Why did Prof. Bianchi win the price?

Tipologia di risposta

E' il tipo che definisce la classe degli oggetti richiesti dalla domanda. Non esiste una lista precisa di tali tipi poiché essi coinvolgono direttamente la conoscenza del sistema. Sul piano concettuale tali classi potrebbero essere riferite a tutti, o quasi, i predicati presenti nella base di conoscenza, o alle classi definite in una ontologia. In generale non esiste neppure una metodologia largamente accettata per definire questi tipi, ma col tempo è diventata prassi comune definire alcune categorie generali di riferimento:

Focus e tema della domanda

Il focus della domanda è tipicamente composta da una frase nominale che descrive la proprietà o l'entità da cercare per rispondere alla domanda. Questo è usualmente interno all'espressione di domanda. Tuttavia si noti che se da

Tabella 5.2. Elenco delle tipologie di risposta più comuni presentata da Moldovan in[109]

Quesion Class	Question	Answer Type
WHAT		Money, Number, Definition, Title, NNP, Undefined
WHO		Person
WHY		Explanation/Reason
WHOM		Person
HOW	basic-how	Manner
	how-many	Number
	how-long	Time/Distance
	how-much	Money/Price
	how-much	Undefined
	how-far	Distance
	how-tall	Number
	how-rich	Undefined
	how-large	Number
WHERE		Location
WHEN		Date
WHICH	which-who	Person
	which-where	Location
	which-when	Date
	which-what	NNP
NAME	name-who	Person
	name-where	Location
	name-what	Title/NNP

un lato il focus influenza il tipo assegnato alla riposta, dall'altro esso rimane comunque una proprietà esclusiva della domanda che può non trovare riscontro nella base di conoscenza. Nella domanda "*What composer wrote the Tosca ?*", ad esempio, il focus è il compositore ma il tipi associabile dal sistema potrebbe limitarsi ad identificarlo con il tipo *Person*.

Il tema della domanda si riferisce invece all'oggetto che la proposizione principale della domanda discute. Nella domanda precedente "*What composer wrote the Tosca ?*" il tema è *the Tosca* mentre il focus è riferito ad una sua proprietà *composer*.

Risposta candidata

Una risposta candidata, nel contesto definito da una domanda, è una piccola porzione di testo, tipicamente identificata con una frase nominale, che il sistema valuta come idonea a rispondere alla richiesta. Il tipo associabile dal sistema a questa risposta deve essere conforme al tipo individuato in fase di analisi della domanda. Può capitare tuttavia che tale controllo sia in qualche

modo rilassato per conferire una maggiore robustezza al sistema a scapito di una leggera perdita di precisione.

5.4 Architettura

I primi tentativi di costruire un sistema QA erano fondati sull'esistenza su grandi collezioni di informazione testuale non strutturata. Benché più contenute rispetto alle informazioni contenute nel Web, queste collezioni, rappresentavano comunque una serie di problematiche sul piano computazionale nonostante fossero caratterizzate da un lessico uniforme ed da informazioni consistenti e curate. Da ciò il proliferare di approcci basati su tecnologie impiegate nei sistemi di recupero dell'informazione.

Tradizionalmente l'approccio più diffuso prevedeva l'utilizzo di motori di ricerca per ricercare i termini estratti dalla domanda all'interno di una collezione di documenti. Tralasciando di fatto aspetti sintattici e semantici, le interrogazioni verso i motori di ricerca conteneva sole le parole-chiave selezionate fra quelle presenti nella domanda. Tali termini erano sottoposti a tecniche di normalizzazione e di allineamento con gli indici del motore di ricerca al fine di aumentare le probabilità di recupero. I documenti ritornati, ordinati una prima volta in base ai criteri interni al motore di ricerca, subivano poi un'ulteriore fase di riordinamento in base alla loro vicinanza con i termini della domanda. Un esempio di tale architetture è riportato in figura 5.1.

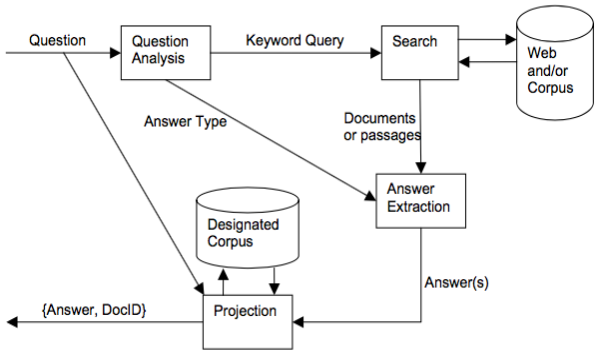


Figura 5.1. Prototipo di sistema QA definito in [122].

Seguendo questo schema, la domanda è prima analizzata per poter estrarre parole-chiave e definirne il tipo. I termini estratti sono poi utilizzati dal modulo di ricerca per ritornare l'insieme dei documenti, o delle porzioni di testo indicizzate, più idonei a costruire una risposta alla domanda. Infine il tipo associato alla domanda è invece impiegato nel modulo denominato *answer extraction* per ordinare i documenti ritornati.

5.4.1 Analisi della domanda

Analizzare una domanda impone implicitamente un insieme di assunzioni che riguardano maggiormente la definizione di un insieme di tipi sui quali operare le scelte sulle strategie da utilizzare.

Nella realizzazione di un sistema QA una questione molto importante riguarda il dominio(generico o specifico) sul quale è possibile ricevere una risposta. Selezionato un dominio di interesse, il primo aspetto sul quale porre attenzione è il livello di genericità con il quale si vogliono trattare le sue informazioni. Tipicamente in base alle domande più frequenti si definisce un sistema di entità così da catturare(ed organizzare) i concetti che compaiono all'interno di tali interrogazioni. Successivamente si definiscono i meccanismi di recupero necessari in presenza di entità sconosciute, la cui logica è spesso influenzata dal dominio in esame. Tuttavia nel costruire un sistema di risposta *aperto* non è possibile assumere l'esistenza di una definizione completa delle entità universali. Non esiste infatti nessun principio guida nella definizione delle proprietà di questo insieme; ne tanto meno i molti tentativi presenti in letteratura offrono qualche preferenza verso una teoria rispetto ad un'altra per poter definire la natura degli elementi di questo insieme. Ciononostante nel passato un insieme simile ai tipi riporti in 5.2 è stato spesso utilizzato, ma ciò ha però evidenziato come fosse necessaria di volta in volta integrare ulteriore informazione di dominio per rispondere correttamente alle richieste.

Generalmente quindi l'analisi della domanda seleziona l'entità più idonea a rappresentare l'unità nominale all'interno del sistema. Se è possibile selezionare un'entità fra quelle presenti nel sistema, si potranno allora impiegare le procedure ad essa associate per definire il suo contributo semantico. Nel caso in cui ciò non fosse possibile, allora l'espressione linguistica riceverà un valore semantico neutro o di default. Quest'ultimo tuttavia avrà una bassa probabilità di contribuire alla costruzione della risposta.

Quanto appena detto sottolinea maggiormente come sia in generale complesso affrontare il problema di selezione del giusto insieme di entità. Soprattutto è facile intuire come sia necessario garantire la copertura della maggior parte dei casi attesi per permettere una reale applicazione del sistema QA.

5.4.2 Classificazione della domanda

Il processo di classificazione è tipicamente basato su regole che definiscono un insieme di schemi così da assegnare una determinata entità a particolari espressioni linguistiche. La presenza, ad esempio, di espressioni come {*who,whom,whose*} suggerisce facilmente la definizione dell'entità *Person* come tipo da assegnare alla risposta. Generalmente questi schemi sono caratterizzati da un alto livello di accuratezza, ma da una bassa di capacità di recupero in presenza di entità generiche. Infatti più alto è il grado di specificità nella definizione di un'entità, maggiore sarà la probabilità che lo schema

associato copra bene i casi d'interesse mentre non venga applicato nei casi più generali.

Nei casi in cui non sia possibile selezionare uno schema da applicare, la politica del sistema potrebbe prevedere il rifiuto della domanda. Attualmente non si conosce nessun tipo di soluzione generale da applicare quando le domande non sono coperte dagli schemi del sistema. Tuttavia è prassi consolidata restringere gli eventuali recuperi ai soli casi in cui lo schema riferito ad una parola *Wh-* non riesca a risolvere l'entità ad essa assegnata che svolge il ruolo di focus della domanda, come accade in "*What X ...*" . In tali situazioni l'approccio più utilizzato prevede l'impiego di conoscenza aggiuntiva, solitamente in forma di ontologie, per aumentare le possibilità di risoluzione dell'entità in esame.

5.4.3 Ricerca delle informazioni

Originariamente le parole-chiave estratte durante l'analisi della domanda erano utilizzate per eseguire ricerche su collezioni di documenti rappresentanti il dominio. Prima della diffusione delle più recenti reti semantiche, o grafi di conoscenza, l'approccio tipicamente utilizzato nei sistemi QA era basato sui motori di ricerca o sistemi di recupero dell'informazione. Tuttavia il classico processo di selezione operato da questi sistemi era spesso non sufficiente ad individuare i documenti necessari per la risposta.

L'approccio denominato *annotazione predittiva* rientra fra le proposte rivolte all'incremento della precisione e della capacità di recupero dei sistemi impiegati. Secondo quest'approccio l'indice del motore di ricerca doveva essere arricchito con informazioni aggiuntive per garantire che ogni passaggio selezionato contenesse almeno un candidato per la risposta. La meta-informazione da inserire all'interno dell'indice del motore di ricerca non era costituita da nuovi termini, come avviene nella tecnica di espansione della domanda, ma da informazioni relative al tipo del documento, tipicamente implicite nella struttura dell'indice. Si consideri ciò che accade con la rimozione delle parole "*Wh-*" dagli elementi della domanda. Il sistema di recupero non potrà sfruttare le utili informazione ad esse associate per selezionare il documento più pertinente. Nel caso specifico della rimozione della parola *when*, ad esempio, il sistema non potrà essere guidato in nessun modo verso la selezione di passaggi contenenti informazioni di tipo temporale. Al contrario se si elabora la collezione di documenti con il risolutore di entità prima del processo di indicizzazione, allora le classi semantiche estratte potranno essere impiegate come meta informazione all'interno dell'indice per caratterizzare meglio il documento.

Rispetto ai classici sistemi di risposta l'utilizzo dell'annotazione predittiva richiede due operazioni aggiuntive. La prima, come già accennato, riguarda l'annotazione del corpo dei documenti con l'informazione di carattere semantico prodotta dal risolutore di entità. La seconda richiede invece che il tipo di risposta individuato durante la fase di analisi sia incluso negli elementi

che costituiranno la richiesta al motore di ricerca. In questo modo è possibile ottenere un numero di passaggi, o di documenti, inferiore rispetto alla metodologia classica.

Si consideri ad esempio la domanda "*Where is New York located?*". Qui la parola *where* può essere potenzialmente associata a diverse entità, tutte sotto-categorie di *Place* come {*Continent, WorldRegion, Country, State, City, Capital, Lake, River...*}. Assumendo poi che il processo di risoluzione delle entità associ l'entità *City* all'elemento "*New York*", che esista un'euristica associata all'espressione "*where-is*" che selezioni l'entità contenete il contenuto della domanda, che esista una relazione di contenimento definita sulle sotto-categorie di *Place*, è allora possibile selezionare i tipi di risposta: {*Continent, WorldRegion, Country, State*}.

Si noti tuttavia come sia necessario nell'includere, all'interno del processo di indicizzazione anche le entità del sistema, porre particolare attenzione alla gestione della loro gerarchia. Nei casi in cui le domande abbiano una forma simile a "*What city... ?*", allora la restrizione sulle sole entità associate con *City* può essere raggiunta semplicemente annotando tutte le città conosciute con l'entità *City*, indicizzando tale annotazione e inserendo infine *City* nell'interrogazione verso il sistema di recupero. Ma in domande meno restrittive, come "*Where was John Lennon killed?*", non è possibile associare nulla oltre l'entità più generale attribuita a *where*, cioè *Place*. Per poter comunque ritornare l'entità desiderata, *New York City*, l'approccio delle annotazioni predittive prevede tre possibili soluzioni:

- l'interrogazione verso il motore di ricerca è estesa con una disgiunzione di tutti i possibili sotto-tipi dell'entità selezionata.
- Il controllo di sussunzione fra le entità candidate e l'entità selezionata è eseguito dal motore di ricerca.
- Tutte le entità sono simultaneamente annotate ed indicizzate con tutte le entità da loro implicate.

5.5 Complessità di una domanda

Finora si è accennato alle difficoltà relative alla definizione di cosa sia una domanda. Anche se non esplicitamente espresso, tutte le interrogazioni trattate fanno tutte parte di una stessa classe. Tale classe può essere definita come l'insieme di tutte le domande le cui risposte possono essere costruite a partire da qualche meccanismo di ricerca dell'informazione contenuta in una base di conoscenza. Esistono tuttavia domande, come quelle riportate nel seguito, che pur non essendo particolarmente difficili nella realtà non possono essere risposte con la questa metodologia. Si considerino le seguenti domande:

- What is one plus one?
- How many legs does a dog have?
- How many books are there (approximately) in a local library?

- What was the dilemma facing Hamlet?

Ognuna delle precedenti domande è caratterizzata da una particolare richiesta: sia essa di natura matematica, logica, di semplice stima, o d'interpretazione. Risolvere ognuna di queste richieste, impiegando la stessa tecnica utilizzata in precedenza, imporrebbe l'assunzione che l'informazione necessaria a costruire la risposta sia contenuta all'interno della base di conoscenza. Sfortunatamente, come discusso nel capitolo riguardante la semantica del linguaggio, un tale approccio è destinato a fallire non solo per l'impresa infattibile di enumerare la realtà, ma soprattutto perché distante dal principio di sistematicità del linguaggio che guida la sua interpretazione. Quindi per fornire una risposta corretta alle domande riportate sopra la sola conoscenza di informazione non è sufficiente, ma occorre equipaggiare il sistema QA di una forma di ragionamento più avanzata in grado di costruire delle giustificazioni a supporto delle risposte generate.

Per chiarire questo punto si consideri la domanda d'esempio "*What color does litmus paper turn when it comes into contact with a strong acid?*". Per rispondere a tale domanda si può assumere l'esistenza di un qualche documento contenente il seguente passaggio "*Litmus is the compound that turns red in acid solution.*". Sebbene attraverso la comune capacità interpretativa è facile definire come risposta alla domanda precedente l'entità *red*, il passaggio selezionato non è da solo sufficiente a fornire una risposta esaustiva alla domanda posta. Detto in altri termini, l'esempio appena mostrato evidenzia la necessità di impiegare una certa quantità di conoscenza, che può essere associata al senso comune, all'interno del processo d'inferenza.

Nell'esempio in corso, infatti," *Litmus paper*" in quanto cartina deve essere assimilato ad un generico foglio di carta. Ma "*paper*", essendo parte di un'espressione nominale composta, riceve un insieme di caratteristiche derivanti dall'elemento di testa "*Litmus*". Qui il primo problema da trattare riguarda la conoscenza dell'esatta misura di similarità che sussiste fra l'entità *Litmus paper* e l'entità *Litmus*. La seconda problematica da trattare riguarda invece l'interpretazione di "*strong acid*", contenuto nella domanda, ma che non trova un esatto equivalente all'interno del passaggio selezionato dal sistema come candidato a supporto della risposta. Se nell'esempio in questione non fa molta differenza il livello di acidità della soluzione ai fini della colorazione della cartina tornasole, in generale non è così facile definire questo tipo di conoscenza e rendere equivalenti le due entità "*strong acid*" e "*acid solution*".

Rimane infine il problema di esplicitare lo stato di *contatto* che caratterizza gli elementi presenti in una soluzione, ma quest'ultimo tipo di informazione può essere ricondotto a una qualche forma di ragionamento di senso comune e magari ricavato utilizzando qualche risorsa esterna.

Sulla base delle considerazioni appena trattate, è opportuno identificare alcuni modelli di ragionamenti capaci di semplificare il processo di risposta. Fra questi la possibilità di adottare un approccio composizionale per la definizione della semantica della domanda può essere sicuramente interessante. Tuttavia

rimane da valutare se il ricorso ad un tale approccio sia sempre necessario. In presenza di informazione incompleta infatti è preferibile tralasciare i costrutti della domanda che non ricevono una denotazione e proseguire l'analisi con la conoscenza assegnata ai restanti costrutti. Ma è anche vero il contrario: anche semplici domande come "*Who is the wife of the President?*" sono più facili da trattare se decomposte in una serie di sotto-domande. Nel caso specifico riconosce quale entità sia la denotazione della parola *President* facilita sicuramente il processo di risposta.

In generale altro aspetto problematico riguarda l'eccessiva specificazione che può caratterizzare alcune domande. Per introdurre la nozione di eccessiva specificazione è preferibile utilizzare alcuni esempi. Nella domanda "*Who coined the term cyberspace in his novel Neuromancer ?*" il ruolo della frase preposizionale "*in his novel Neuromancer*" è semplicemente descrittivo poiché non esistono due entità distinte che hanno coniato il termine "*cyberspace*" e sono autori del romanzo "*Neuromancer*". Ciononostante la domanda suggerisce un vicolo di uguaglianza da imporre sulle entità presenti rispettivamente nei predicati per chi ha coniato il termine e per chi ha scritto il romanzo. Per trattare questo genere di casi si possono definire i seguenti principi:

- Analisi della configurazione : alcune proprietà strutturali delle domande sono indipendenti dall'entità coinvolte. L'insieme di domande riguardanti fatti o eventi tende il più delle volte ad essere caratterizzato dalla stessa costruzione. Tuttavia la definizione degli schemi di domanda dovrebbe essere fatta con cautela per non incidere negativamente con le prestazioni del sistema.
- Rilassamento dei vincoli : se non sia possibile conoscere a priori la funzione dei modificatori(descrittivi o restrittivi), è opportuno utilizzarli tutti. Nel caso in cui l'insieme di risposta risulti vuoto si può tentare la rimozione di qualche vincolo. Tuttavia occorre sottolineare come a volte una risposta vuota può essere corretta, e che quindi non si debba sempre rimuovere vincoli.
- Verifica della risposta : indipendentemente dalle entità selezionate per rappresentare le entità della domanda occorre sempre garantire che la loro composizione si logicamente valida.

5.6 Sommario e ulteriori riferimenti

Il primo sistema QA presentato in letteratura fu *baseball* porposto da Green e colleghi in [55]. Dopo quasi un ventennio fu presentato *lunar* in [149]. Informazioni sul sistema *START*[1], sviluppato da Boris Katz, possono essere trovate in [76], mentre i riferimenti per i sistemi *lasso* e *falcono* sono rispettivamente [109] e [60]. Gaizauskas e Humphreys presentarono invece il sistema

[1] pubblicamente disponibile a http://start.csail.mit.edu/

QA-LaSIE in [50]. I tentativi di applicare tecniche di apprendimento automatico ai sistemi QA sono poi documentati nei lavori di Zhang e Lee [151, 150]. Infine il sistema Watson fu presentanto publicamente nel lavoro [43].

Gli argomenti e la discussione condotta in questo capitolo è largamente ispirata dal lavoro di Prager [122] sui sistemi di risposta per domini aperti. Dagli argomenti trattati è facile concludere che la costruzione di un sistema QA sia in generale una faccenda abbastanza complessa. Sebbene vi siano ormai delle metodologie ben definite, risolvere le problematiche relative alla conoscenza generica e al comune senso di ragionamento richiede ancora ulteriore investigazione. Nel prossimo capitolo sarà presentato un tentativo di risolvere alcuni dei problemi discussi qui sfruttando un approccio di ragionamento dichiarativo basato su programmi logici.

6

ASP : un formalismo per rappresentare e ragionare sulla conoscenza

Rappresentare e manipolare informazioni di tipo complesso richiede un adeguato formalismo logico. Oggigiorno la maggior parte delle basi di conoscenza pubblicamente disponibili, come Wikidata, il Knowledge Graph di Google etc., adottano formalismi basati su logiche più o meno complesse.

Fra i formalismi logici presenti in letteratura la programmazione logica disgiuntiva (DLP) è sicuramente una scelta solida, poiché la ricerca e gli strumenti ad essa associati hanno raggiunto un ragguardevole grado di maturità, che ne permette l'uso anche in applicazioni reali di larga scala.

Lo scopo di questo capito è quello di introdurre i concetti di base riferiti alla sintassi di DLP ed alla sua semantica in termini di *Answer-set Programming (ASP)*. Sarà presentata una metodologia di programmazione semplice ed efficace e saranno presentati alcuni esempi di rappresentazione in ASP di problemi computazionalmente difficili, in modo da evidenziare le capacità espressive del linguaggio.

6.1 Programmazione logica disgiuntiva

I programmi logici disgiuntivi sono programmi logici nei quali è permesso avere disgiunzione nella testa e negazione nel corpo delle regole. Numerosi lavori in letteratura ne evidenziano la capacità di esprimere forme di ragionamento non monotono attribuibili al senso comune. Recentemente sono state anche considerate opportune estensioni del linguaggio base che consentono di modellare naturalmente anche situazioni in cui non è possibile assumere un'informazione completa.

La più accettata definizione di semantica è riferita alla proposta di Gelfond e Lifschitz, conosciuta con il nome di *answer sets semantics* []. In questa definizione di semantica, un programma logico disgiuntivo può avere, non è detto che esistano, diversi modelli alternativi, chiamati *answer sets* (o *insiemi di risposta*, in italiano), che rappresentano le diverse possibili "viste" del mondo. DLP, con semantica definita sugli insiemi di risposta, risulta essere

un formalismo molto espressivo capace di rappresentare i problemi al secondo livello della gerarchia polinomiale, in particolare tutti i problemi appartenenti alla classe di complessità Σ_P^2. Ricordiamo che Σ_P^2 è la classe dei problemi decidibili in tempo polinomiale da una macchina di Turing non deterministica con un *oracolo* in *NP*, mentre Π_P^2 è la classe dei problemi i cui complementi sono in Σ_P^2.

L'alta espressività permette di rappresentare molti rilevanti problemi di natura pratica. Ovviamente il prezzo da pagare per tale espressività è dato dagli alti costi computazionali, nel caso peggiore. Più in dettaglio la complessità per le forme di ragionamento *brave* e *cautious* nei casi di programmi con disgiunzione è rispettivamente Σ_P^2 e Π_P^2. Per i programmi senza disgiunzione tale complessità scende a *NP* e *Co-Np*. Il sistema DLV [], frutto di un progetto congiunto dell'università della Calabria e dell'università di Vienna, fornisce una implementazione efficiente del linguaggio completo DLP, oltre che prevedere numerose estensioni richieste negli ultimi anni per le varie applicazioni concrete nelle quali è stato impiegato.

Dopo il suo esordio nel 1997, il sistema è stato continuamente migliorato ed arricchito nel corso degli anni. Poiché le finalità del progetto erano soprattutto orientate all'efficienza, molte tecniche di ottimizzazione sono state integrate in quasi tutti i moduli: veloci generatori di istanze, moderne tecniche di controllo degli insiemi di risposta, euristiche per la generazione dei modelli etc. Attualmente il sistema DLV rappresenta una delle implementazioni più efficienti per DLP, ed è ampiamente usato in ambito accademico e industriale.

6.2 Core Language

Il linguaggio utilizzato da DLV è una variante di *datalog* disgiuntivo, senza simboli di funzione, interpretato secondo la semantica degli insiemi di risposta (answer set) ed esteso con la definizione di vincoli deboli (*weak constraint*). Questi ultimi rappresentano una generalizzazione dell'idea di vincolo standard (*hard constraint*), poiché prevedono una misura di violazione associata a ciascuna istanza di vincolo che viene utilizzata per modellare priorità e problemi di ottimizzazione.

Nel presentare il linguaggio si segue la convezione riferita a *Prolog*, secondo la quale stringhe che iniziano con una lettera maiuscola denotano una variabile, mentre quelle che hanno una lettera minuscola come loro primo carattere rappresentano una costante. In DLV è poi possibile usare numeri interi e stringhe di testo arbitrarie racchiuse fra virgolette.

6.2.1 Sintassi

Un *termine* è o una variabile o una costante. Un *atomo* è un'espressione $p(t_1,\ldots,t_n)$, dove p è un *predicato* di arità n e $t_1,\ldots,t_n$ sono termini. Un *classico letterale* l è o un atomo p (nel qual caso è detto *positivo*), o un atomo

negato $\neg p$ (detto *negativo*). Un letterale *negato per fallimento (NAF)* ℓ è della forma l o not l, dove l è un classico letterale; nel primo caso ℓ è *positivo*, e nel secondo *negativo*. Nel seguito, se non specificato diversamente, con il termine *letterale* si farà riferimento sempre a letterali classici.

Dato un letterale l, il suo *complemento* $\neg l$ è definito come $\neg p$ se $l = p$ e p se $l = \neg p$. Un insieme L di letterali è detto essere consistente se, per ogni letterale $l \in L$, il suo complemento non è contenuto in L.

Oltre ai normali predicati che si possono definire, DLV fornisce alcuni predicati speciali utili per eseguire confronti (=, <, >, <>), e per definire semplici operazioni aritmetiche come somma e prodotto.

Una *regola disgiuntiva* r è una formula:

$$a_1 \ \mathtt{v} \ \cdots \ \mathtt{v} \ a_n \ \mathtt{:-} \ b_1, \cdots, b_k, \ \text{not } b_{k+1}, \cdots, \ \text{not } b_m. \tag{6.1}$$

dove $a_1, \cdots, a_n, b_1, \cdots, b_m$ sono letterali classici e $n \geq 0$, $m \geq k \geq 0$. La disgiunzione $a_1 \ \mathtt{v} \ \cdots \ \mathtt{v} \ a_n$ è la *testa* di r, mentre la congiunzione $b_1, ..., b_k, \ \text{not } b_{k+1}, ..., \ \text{not } b_m$ è il *corpo* di r. Una regola senza letterali in testa ($n = 0$) è usualmente riferito come un *vincolo d'integrità*. Una regola avente precisamente un solo letterale in testa ($n = 1$) è denominata come una *regola normale*. Se infine il corpo è vuoto ($k = m = 0$), allora la regola è indicata come *fatto* della base di conoscenza e sarà omesso il segno " :- ".

Se r è una regola di forma simile a (6.1), allora $H(r) = \{a_1, \ldots, a_n\}$ è l'insieme di letterali nella testa e $B(r) = B^+(r) \cup B^-(r)$ è l'insieme di letterali del corpo, dove $B^+(r)$ indica la parte *positiva del corpo* ed è composta da $\{b_1, \ldots, b_k\}$ e $B^-(r)$ rappresenta invece la parte negativa.

Un programma logico disgiuntivo $\mathcal{P}$ è un 'insieme finito di regole. Un programma $\mathcal{P}$ senza "not" (ad esempio tale che $\forall r \in \mathcal{P} : B^-(r) = \emptyset$) è detto *positivo*[1], mentre un programma $\mathcal{P}$ senza v (tale che $\forall r \in \mathcal{P} : |H(r)| \leq 1$) è detto *programma datalog* (o *programma logico normale*).

Il linguaggio del sistema DLV estende datalog disgiuntivo con un altro costrutto denominato *vincolo debole*(weak constraint). I vincoli deboli possono essere immaginati come una variante dei vincoli d'integrità. A livello sintattico la differenza con quest'ultimi è l'utilizzo del simbolo ":∼" invece di " :- ". Tali vincoli offrono la possibilità di specificare esplicitamente un peso o un livello di priorità.

Formalmente, un vincolo debole wc è un'espressione della seguente forma:

$$:\sim \ b_1, \ldots, b_k, \ \text{not } b_{k+1}, \ldots, \ \text{not } b_m. \ [w : l]$$

dove $m \geq k \geq 0$, $b_1, \ldots, b_m$ sono letterali classici, mentre w (il *peso*) e l (il *livello*) sono interi positivi o variabili. Nel caso w e/o l siano omessi, sono da considerarsi come equivalenti ad 1. Gli insieme $B(wc)$, $B^+(wc)$, e $B^-(wc)$ di un vincolo debole wc sono definiti nello stesso modo dei vincoli d'integrità tradizionali.

[1] In un programma positivo la negazione per fallimento (not) non occorre, mentre la negazione forte (o classica ¬) può essere presente.

Un programma DLV $\mathcal{P}$ è un insieme finito di regole (possibilmente contenente vincoli d'integrità) e vincoli deboli. Per un programma $\mathcal{P}$, sia $WC(\mathcal{P})$ l'insieme di vincoli deboli contenuti in $\mathcal{P}$, e sia $Rules(\mathcal{P})$ l'insieme delle regole(comprensive dei vincoli d'integrità) in $\mathcal{P}$.

Una regola è *sicura* se ogni sua variabile appare in almeno un letterale positivo nel corpo, ad esclusione dei predicati speciali di comparazione. Un programma è sicuro, se ognuna delle sue regole è sicura. Nel seguito saranno considerati solo programmi sicuri.

Infine, un termine sia esso un atomo, una regola, o un programma, è detto *fissato*(ground), se al suo interno non compaiono variabili. Un programma di questo tipo è spesso riferito come programma proposizionale.

6.2.2 Semantica

Al fine di gestire correttamente i vincoli deboli la semantica dei programmi DLV estende la semantica basata su insiemi di risposta per i programmi datalog disgiuntivi, originariamente definita in [53]. La semantica proposta è una piccola generalizzazione rispetto al lavoro di [15] per permettere una strutturazione a livelli dei pesi associati a tali vincoli.

Universo di Herbrand

Per un qualsiasi programma $\mathcal{P}$, sia $U_{\mathcal{P}}$ (l'Universo di Herbrand) l'insieme di tutte le costanti che appaiono in $\mathcal{P}$. Nel caso non vi siano costanti in $\mathcal{P}$, allora una costante arbitraria è artificialmente aggiunta $U_{\mathcal{P}}$.

Base di letterali di Herbrand

Per un qualsiasi programma $\mathcal{P}$, sia $B_{\mathcal{P}}$ l'insieme di tutti i predicati fissati costruibili dai simboli di predicato in $\mathcal{P}$ sostituendo al loro interno le variabili con le costanti in $U_{\mathcal{P}}$(si noti che, per ogni atomo p, $B_{\mathcal{P}}$ contiene anche i letterali negati classicamente $\neg p$).

Istanziazione fissata

Per una qualsiasi regola r, con $Ground(r)$ si indica l'insieme di regole ottenute dall'applicazione di tutte le possibili sostituzioni σ della variabili in r con elementi di $U_{\mathcal{P}}$. In modo simile, dato un vincolo debole w, $Ground(w)$ denota l'insieme di vincoli deboli ottenuti dalle possibili sostituzioni σ della variabili in w con elementi di $U_{\mathcal{P}}$. Infine per un qualsiasi programma $\mathcal{P}$, $Ground(\mathcal{P})$ è l'insieme $GroundRules(\mathcal{P}) \cup GroundWC(\mathcal{P})$, dove:

$$GroundRules(\mathcal{P}) = \bigcup_{r \in Rules(\mathcal{P})} Ground(r)$$

e

$$GroundWC(\mathcal{P}) = \bigcup_{w \in WC(\mathcal{P})} Ground(w)$$

Si noti che per i programmi proposizionali, è vera la seguente relazione:

$$\mathcal{P} = Ground(\mathcal{P})$$

Insiemi di risposta

Per un qualsiasi programma $\mathcal{P}$, si definiscono i suoi insiemi di risposta usando la sua istanziazione $Ground(\mathcal{P})$ attraverso un approccio a tre passi:

1. si definiscono gli insiemi di risposta per i programmi datalog disgiuntivi positivi;
2. si *riducono* i programmi che contengono negazione per fallimento nel loro equivalente positivo, e si usano questi programmi per definire gli insiemi di risposta;
3. si specifica il modo in cui i vincoli deboli condizionino la semantica, definendo quindi la semantica di un programma DLV generale.

Un'interpretazione I è un insieme di classici letterali fissati, cioè $I \subseteq B_{\mathcal{P}}$ per un programma $\mathcal{P}$. Un'interpretazione consistente $X \subseteq B_{\mathcal{P}}$ è definita *chiusa sotto* $\mathcal{P}$ se per ogni $r \in Ground(\mathcal{P})$ $H(r) \cap X \neq \emptyset$ per qualsiasi $B(r) \subseteq X$ e $\mathcal{P}$ è un programma datalog disgiuntivo positivo.

Un'interpretazione $X \subseteq B_{\mathcal{P}}$ è un'*insieme di risposta* per $\mathcal{P}$, se esso è minimale(rispetto alla nozione di inclusione insiemistica) fra tutti le (consistenti) interpretazioni che sono chiuse rispetto a $\mathcal{P}$.

Example 6.1. Il programma positivo $\mathcal{P}_1 = \{a \vee \neg b \vee c.\}$ *ha come insiemi di risposta* $\{a\}$, $\{\neg b\}$, *e* $\{c\}$. *La sua estensione* $\mathcal{P}_2 = \{\ a \vee \neg b \vee c.\ \ ;\ \ :\!-a.\ \}$ *ha i seguenti insiemi di risposta* $\{\neg b\}$ *and* $\{c\}$. *Infine, il programma positivo* $\mathcal{P}_3 = \{a \vee \neg b \vee c.\ \ ;\ \ :\!-a.\ \ ;\ \ \neg b :\!- c.\ \ ;\ \ c :\!- \neg b.\}$ *ha un solo insieme di risposta* $\{\neg b, c\}$.

Il *ridotto*, anche detta *trasformata di Gelfond-Lifschitz*, di un programma fissato $\mathcal{P}$ in relazione ad un insieme $X \subseteq B_{\mathcal{P}}$ è il programma positivo fissato $\mathcal{P}^X$, ottenuto da $\mathcal{P}$ attraverso:

- la rimozione di tutte le regole $r \in \mathcal{P}$ per la quale $B^-(r) \cap X \neq \emptyset$ è vera;
- la rimozione della parte negativa dal corpo delle regole rimanenti.

Un insieme di risposta di un programma $\mathcal{P}$ è un insieme $X \subseteq B_{\mathcal{P}}$ tale che X sia un insieme di risposta di $Ground(\mathcal{P})^X$.

Example 6.2. Sia dato un programma generico $\mathcal{P}_4 = \{a \vee \neg b :\!- c.\ \ ;\ \neg b :\!-$ not $a,$ not $c.\ \ ;$ $a \vee c :\!-$ not $\neg b.\}$ *e* $I = \{\neg b\}$, *il ridotto* $\mathcal{P}_4^I$ *è* $\{a \vee \neg b :\!- c.\ \ ;\ \ \neg b.\}$. *Si può facilmente verificare che* I *sia un un insieme di risposta per* $\mathcal{P}_4^I$, *e di conseguenza lo è anche per* $\mathcal{P}_4$.

Si consideri ora $J = \{a\}$. La sua forma ridotta $\mathcal{P}_4^J$ è $\{a \vee \neg b \text{ :- } c. \ ; \ a \vee c.\}$ e J è un insieme di risposta per $\mathcal{P}_4^J$, così da renderlo un insieme di risposta anche per $\mathcal{P}_4$.

Se, invece, si considera $K = \{c\}$, il suo ridotto $\mathcal{P}_4^K$ è uguale a $\mathcal{P}_4^J$, ma K non è un insieme di risposta per $\mathcal{P}_4^K$: per la regola $r : a \vee \neg b \text{ :- } c$, $B(r) \subseteq K$ risulta vera, ma $H(r) \cap K \neq \emptyset$ no. Quindi, può essere verificato che I e J siano i soli insiemi di risposta per $\mathcal{P}_4$.

Remark 6.3. In alcuni casi, è tuttavia possibile simulare la disgiunzione attraverso l'uso di regole normali non stratificate. Seguendo la tecnica descritta in [4, 34, 85] si può effettivamente spostare la disgiunzione nel corpo della regola come mostrato dall'esempio seguente. Si consideri $\mathcal{P}_5 = \{ \ a \vee b. \ \}$ e la sua versione modificata $\mathcal{P}_5' = \{ \ a \text{ :- not } b. \ ; \ b \text{ :- not } a. \ \}$. Entrambi i programmi sono caratterizzati dagli stessi insiemi di risposta: $\{a\}$ and $\{b\}$.

Tuttavia tale tecnica non è applicabile in generale. Si consideri, ad esempio, $\mathcal{P}_6 = \{ \ a \vee b. \ ; \ a \text{ :- } b. \ ; \ b \text{ :- } a. \ \}$. Questo programma ha $\{a, b\}$ come suo unico insieme di risposta, mentre la sua versione modificata $\mathcal{P}_6' = \{ \ a \text{ :- not } b. \ ; \ b \text{ :- not } a. \ ; \ a \text{ :- } b. \ ; \ b \text{ :- } a. \ \}$ non ne possiede nessuno.

Tali considerazioni attestano come $\mathcal{P}_5$ e $\mathcal{P}_5'$ non siano "fortemente equivalenti" come definito in [89]. Tuttavia non vi è nessuna evidente relazione fra i programmi modificati con la tecnica precedenti e la nozione di equivalenza in senso forte. Infatti i programmi possono essere fortemente equivalenti, come nel caso di $\mathcal{P}_5 \cup \{ \text{ :- } a, b.\}$ e $\mathcal{P}_5' \cup \{ \text{ :- } a, b.\}$), equivalenti $\mathcal{P}_5$ e $\mathcal{P}_5'$, o non essere equivalenti per nulla, tipo $\mathcal{P}_6$ e $\mathcal{P}_6'$.

Dato un programma fissato $\mathcal{P}$ con vincoli deboli $WC(\mathcal{P})$, si possono richiedere gli insiemi di risposta tale per cui $Rules(\mathcal{P})$ minimizzi la somma dei pesi dei vicoli deboli violati(cioè non soddisfatti) nel livello più alto di priorità[2], e fra loro quelli che minimizzano la somma dei pesi dei vincoli deboli nel livello immediatamente inferiore, e così via. Formalmente quest'idea è espressa attraverso la funzione obiettivo $H^{\mathcal{P}}(A)$ per $\mathcal{P}$ e un insieme di risposta A. Usando una funzione ausiliare $f_{\mathcal{P}}$, che riporta i pesi riferiti ai singoli livelli a valori di priorità su un'unica scala di valori, è possibile definire $H^{\mathcal{P}}(A)$ come:

$$\begin{aligned} f_{\mathcal{P}}(1) &= 1, \\ f_{\mathcal{P}}(n) &= f_{\mathcal{P}}(n-1) \cdot |WC(\mathcal{P})| \cdot w_{max}^{\mathcal{P}} + 1, \quad n > 1, \\ H^{\mathcal{P}}(A) &= \sum_{i=1}^{l_{max}^{\mathcal{P}}} (f_{\mathcal{P}}(i) \cdot \sum_{w \in N_i^{\mathcal{P}}(A)} weight(w)), \end{aligned}$$

dove $w_{max}^{\mathcal{P}}$ e $l_{max}^{\mathcal{P}}$ rappresentano rispettivamente il peso e livello massimo fra i vincoli deboli di $\mathcal{P}$; $N_i^{\mathcal{P}}(A)$ si riferisce all'insieme di vincoli deboli nei livello i che sono violati da A, e $weight(w)$ rappresenta il peso del vincolo w. Si noti che $|WC(\mathcal{P})| \cdot w_{max}^{\mathcal{P}} + 1$ è maggiore della somma di tutti i pesi del

[2] Valori alti dei pesi e dei livelli di priorità indicano l'importanza attribuita al vincolo. Per esempio, i vincoli più importanti sono caratterizzati dai più alti pesi associati al livello di priorità più alto.

programma, e che di conseguenza risulta maggiore della somma dei pesi in ogni singoli livello. Intuitivamente, la funzione $f_{\mathcal{P}}$ gestisce i livelli di priorità garantendo che la violazione di un singolo vincolo del livello di priorità i sia più "costoso" della violazione di tutti i vincoli deboli dei livelli sottostanti.

Per un programma DLV $\mathcal{P}$ (possibilmente con vincoli deboli), un insieme A è un insieme di risposta ottimale per $\mathcal{P}$ se e solo s A è un insieme di risposta di $Rules(\mathcal{P})$ e $H^{\mathcal{P}}(A)$ è il l'insieme di risposta minimale fra tutte le $Rules(\mathcal{P})$.

Example 6.4. Si consideri il seguente programma $\mathcal{P}_{wc}$ contenente tre vincoli deboli:

$a \vee b.$
$b \vee c.$
$d \vee \neg d$:- $a, c.$
$:\sim b.$ $[1:2]$
$:\sim a, \neg d.$ $[4:1]$
$:\sim c, d.$ $[3:1]$

$Rules(\mathcal{P}_{wc})$ ammette tre insiemi di risposta : $A_1 = \{a, c, d\}$, $A_2 = \{a, c, \neg d\}$, e $A_3 = \{b\}$. Si ha allora: $H^{\mathcal{P}_{wc}}(A_1) = 3$,$H^{\mathcal{P}_{wc}}(A_2) = 4$, $H^{\mathcal{P}_{wc}}(A_3) = 13$. Dunque, l'unico insieme di risposta ottimale è $\{a, c, d\}$ con peso 3 nel livello 1 and peso 0 nel livello 2.

6.3 Rappresentazione della conoscenza in DLV

Rispetto ad altri sistemi basati sulla programmazione orientata agli insiemi di risposta, DLV offre come principale vantaggio un più ampio campo applicativo. E' infatti pratica abbastanza comune specializzare un sistema su una specifica classe di problemi, ad esempio quelli NP-complete, al fine di aumentarne l'efficienza nella risoluzione delle istanze di questi problemi. Al contrario DLV è progettato per essere largamente applicabile spaziando da compiti orientati maggiormente alle basi di dati a problemi di ricerca e ottimizzazione di complessità decisamente superiore che risiedono nel secondo livello della gerarchia polinomiale, in particolare Σ_2^P e Δ_3^P.

Per descrivere l'utilizzo di DLV come formalismo per la rappresentazione della conoscenza e ragionamento, si discutono dapprima alcuni problemi più vicini alle basi di dati deduttive, e successivamente si presenterà una metodologia per guidare la codifica di problemi di ricerca e di ottimizzazione in modo altamente dichiarativo.

6.3.1 Applicazioni riferite alle basi di dati deduttive

I primi esempi di utilizzo del sistema DLV si riferiscono a problemi che possono essere codificati usando semplicemente regole logiche positive.

Raggiungibilità

Dato un grafo diretto $G = (N, A)$, si vogliono calcolare tutte le coppie di nodi $(a, b) \in N \times N$ tale che b sia raggiungibile a partire da a attraverso una sequenza non vuota di archi in A. Ciò equivale al calcolo della chiusura transitiva della relazione A.

Nella definizione del problema in DLV, si assume che A sia rappresentata dalla relazione binaria $arc(X, Y)$, dove un fatto $arc(a, b)$ attesta la presenza di un arco da a a b in G. L'insieme dei nodi N non è esplicitamente rappresentato poiché i nodi che appaiono nella chiusura transitiva di A sono implicitamente definiti dai fatti relativi alla relazione.

Il seguente programma calcola una relazione $reachable(X, Y)$ contenente tutti i fatti $reachable(a, b)$, tale che b è raggiungibile da a attraverso gli archi del grafo d'ingresso G:

$$reachable(X, Y) \text{ :- } arc(X, Y).$$
$$reachable(X, Y) \text{ :- } arc(X, U), reachable(U, Y).$$

Coppie di persone della stessa generazione

Si consideri la relazione figlio-genitore espressa con un grafo diretto aciclico. Il problema di ricavare tutte le coppie di persone appartenenti alla stessa generazione può essere definito sulla considerazione che due persone per essere selezionate come coppia devono essere o fratelli o figli di persone appartenenti alla stessa generazione.

Le informazioni d'ingresso sono fornite tramite la relazione $parent(X, Y)$, i cui fatti definiscono il grado di parentela che sussiste fra un genitore X ed un figlio Y. Il problema può quindi essere codificato con il seguente programma che calcola la relazione $samegeneration(X, Y)$ sulla base dei fatti della relazione $parent$:

$$samegeneration(X, Y) \text{ :- } parent(P, X), parent(P, Y).$$
$$samegeneration(X, Y) \text{ :- } parent(P1, X), parent(P2, Y), samegeneration(P1, P2).$$

Si noti come a differenza del linguaggio Prolog, l'ordine delle regole e dei sotto-goal(letterali nel corpo delle regole) non influenza il risultato prodotto. Come altri sistemi ASP, DLV permette quindi una codifica regola-per-regola completamente modulare del problema, dove la semantica di ogni regola è indipendente dal suo contesto.

6.3.2 La metodologia GCO

Seguendo il paradigma di codifica *Guess/Check/Optimize*(**GCO**) è possibile definire nel linguaggio di DLV problemi complessi in modo altamente

dichiarativo. Questo paradigma è un'estensione raffinata della metodologia *Guess&Check* presentata in [37]. **GCO** permette di codificare, e quindi risolvere, molti problemi la cui complessità spazia da Σ_2^P-complete a Δ_3^P-complete. Ciò è possibile grazie alla presenza della disgiunzione nella testa delle regole che rendere possibile una modellazione abbastanza intuitiva di problemi che sono più complessi di NP, e dalla possibilità di separare le parti del programma riferite alla logica del problema da quelle riferite alle informazioni d'ingresso.

Dato un insieme di fatti $\mathcal{F}_I$ che specificano una particolare istanza I di un problema **P**, un programma $\mathcal{P}$ per **P**, costruito seguendo l'approccio **GCO**, è composto dalle seguenti tre parti:

La parte di Guessing : questa parte $\mathcal{G} \subseteq \mathcal{P}$ del programma definisce lo spazio di ricerca tale per cui gli insieme di risposta di $\mathcal{G} \cup \mathcal{F}_I$ rappresentino le possibili soluzioni candidate per I.

La parte di Checking : questa parte, opzionale, $\mathcal{C} \subseteq \mathcal{P}$ ha il compito di scartare le soluzioni candidate che non rappresentano istanze ammissibili I del problema. Gli insiemi di risposta di $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I$ saranno quindi le sole soluzioni ammissibili.

La parte di Optimization : questa parte, opzionale, $\mathcal{O} \subseteq \mathcal{P}$ permette di definire un costo quantitativo di valutazione della soluzione utilizzando i vincoli deboli. Implicitamente richiede la definizione di una funzione obiettivo $f : \mathcal{AS}(\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I) \rightarrow \mathbb{N}$ che assegna ad ogni insieme di risposta $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I$ un numero naturale. La semantica di $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I \cup \mathcal{O}$ ottimizza f rimuovendo quegli insieme caratterizzati da valori minimi di valutazione.

Senza imporre alcuna restrizione su quali regole $\mathcal{G}$ e $\mathcal{C}$ possano contenere, è perfettamente legittimo integrare la parte di controllo nella fase di generazione ed avere di conseguenza l'insieme $\mathcal{G}$ che rappresenta l'intero programma e la parte $\mathcal{C}$ vuota. In generale, tuttavia, entrambe le parti del programma $\mathcal{G}$ e $\mathcal{C}$ rappresentano collezioni arbitrarie di regole la cui definizione è legata principalmente alla complessità del problema in esame.

Problemi in NP e Δ_2^P

Per problemi la cui complessità è NP o, per problemi di ottimizzazione in Δ_2^P, il programma che ne codifica le specifiche seguendo la metodologia **GCO**, composto dalle tre parti appena descritte, ha una chiara suddivisione a livelli:

- $\mathcal{G}$ è composta da regole disgiuntive che generano una soluzione candidata S.
- $\mathcal{C}$ definisce i vincoli d'integrità che garantiscono l'ammissibilità di S.
- $\mathcal{O}$ descrive infine l'insieme dei vincoli deboli.

Ogni livello può definire insiemi di predicati ausiliari necessari alla definizione della logica interna.

La disgiunzione nella testa delle regole definisce lo spazio di ricerca del problema, all'interno del quale l'applicazione di ogni singola regola definisce

un punto di scelta. In questo contesto il ruolo dei vincoli d'integrità è quello di eliminare i rami di ricerca non conformi alle specifiche del problema. Infine, i vincoli deboli in $\mathcal{O}$ introducono un ordinamento modulare sugli insiemi di risposta dando la possibilità di preferire alcune soluzioni rispetto ad altre attraverso la definizione della funzione f.

Problemi oltre Δ_2^P

Per problemi la cui complessità estende i limiti della classe Δ_2^P, ed in particolare per quelli Σ_2^P-complete, lo schema definito nella sezione precedente non è più applicabile. Nel caso in cui sia possibile dimostrare che la disgiunzione sia usata esclusivamente per la generazione delle soluzioni candidate e che il livello nel quale essa risiede non presenti cicli al suo interno, allora è possibile garantire che $\mathcal{G}$ abbia complessità in NP. Da ciò altro aspetto che si può garantire è che un insieme di risposta A di $\mathcal{G} \cup \mathcal{F}_I$ possa essere calcolato in tempo polinomiale, cioè creato in modo non deterministico con un numero polinomiali di passi.

Da quanto appena detto è possibile dimostrare come calcolare l'insieme di risposta per l'intero programma, $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I \cup \mathcal{O}$, richieda un tempo polinomiale avendo a disposizione un oracolo in NP. Si noti tuttavia come applicare lo stesso schema ai problemi Σ_2^P-hard implicherebbe che $\Sigma_2^P \subseteq \Delta_2^P$. Fatto questo ritenuto ampiamente falso. Quindi se un programma risolve un problema Σ_2^P-complete e ha una fase di generazione e controllo, $\mathcal{G}$ e $\mathcal{C}$, di complessità inferiore a Σ_2^P, allora $\mathcal{C}$ deve contenere o regole disgiuntive o interferire in qualche modo con $\mathcal{G}$(in particolare nell'unione $\mathcal{G} \cup \mathcal{C}$ devono formarsi dei cicli). Finora infatti è stato tacitamente assunto che la strutturazione del programma in differenti parti imponga un'assenza di un'influenza reciproca fra le diverse parti del programma. Questo aspetto può essere formalizzato in termini di "uso potenziale" delle relazioni [40], o di condizione di divisione d'insieme [90], e assumendo che l'unione di parti del programma non introduca cicli.

Prima di concludere questa sezione è opportuno sottolineare come la metodologia **GCO** abbia implicazioni positive durante lo sviluppo del programma. La struttura modulare che deriva dall'applicazione di **GCO** permette a chi scrive il programma di procedere in modo incrementale scrivendo dapprima la parte di generazione $\mathcal{G}$, valutando successivamente come $\mathcal{G} \cup \mathcal{F}_I$ correttamente produca l'atteso spazio di ricerca, e infine controllare che gli insiemi di risposta di $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I$ sia effettivamente soluzioni ammissibili del problema. Opzionalmente occorre anche verificare che $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I \cup \mathcal{O}$ produca una soluzione ottima al problema.

6.3.3 Applicazioni reali della tecnica di programmazione GCO

Organizzazione degli esami universitari

Si immagini la necessità di dover organizzare una sessione di esami per alcuni corsi universitari. Assumendo come possibili collocazioni temporali, t_1, t_2, e

t_3 ogni corso dovrebbe vedersi assegnato uno ed uno solo di queste date. Le istanze specifiche I di questo problema sono fornite dagli insiemi $\mathcal{F}_I$ che specificato gli esami da organizzare. Il predicato *exam* è caratterizzato da quattro argomenti: un *identificativo* dell'esame, il *professore* responsabile dell'esame, il *curriculum* al quale l'esame appartiene, ed infine l'*anno* nel quale l'esame deve essere svolto nel curriculum.

Sebbene non vi siano restrizioni sulla disponibilità di aule, e quindi vincoli sul numero di esami per data, l'organizzazione deve rispettare comunque le seguenti specificazioni:

$S1$: due esami aventi lo stesso professore come responsabile non possono avere la stessa data;
$S2$: esami nello stesso curriculum devono possibilmente avere differenti date. Se ciò non fosse possibile per tutti gli esami del curriculum C, si dovrebbe:
- $(S2_1)$: minimizzare la sovrapposizione fra gli esami dello stesso anno di C;
- $(S2_2)$: minimizzare la sovrapposizione fra gli esami di differenti anni di C.

Il problema appena definito può essere codificato in DLV con il seguente programma **GCO** $\mathcal{P}_{sch}$:

$$assign(Id, t_1) \vee assign(Id, t_2) \vee assign(Id, t_3) :- exam(Id, P, C, Y). \quad \} \textbf{ Guess}$$

$$:- assign(Id, T), assign(Id', T), \\ Id <> Id', exam(Id, P, C, Y), exam(Id', P, C', Y'). \quad \} \textbf{ Check}$$

$$:\sim assign(Id, T), assign(Id', T) \\ exam(Id, P, C, Y), exam(Id', P', C, Y), Id <> Id'. \ [:2]$$

$$:\sim assign(Id, T), assign(Id', T) \\ exam(Id, P, C, Y), exam(Id', P', C, Y'), Y <> Y'. \ [:1] \quad \} \textbf{ Optimize}$$

La parte di generazione $\mathcal{G}$ ha una singola regola disgiuntiva che definisce lo spazio di ricerca. Mentre la parte di verifica $\mathcal{C}$ consiste di un solo vincolo d'integrità che scarta gli assegnamenti sulla stessa data di esami mantenuti dallo stesso docente. Gli insiemi di risposta di $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I$ sono esattamente quelli corrispondenti alle soluzioni ammissibili, ovvero tutte le soluzioni che soddisfano il vincolo $S1$.

Infine la parte di ottimizzazione, $\mathcal{O}$, è definita da due vincoli deboli con differenti priorità. Entrambi questi vincoli definiscono che esami appartenenti allo stesso curriculum dovrebbero essere svolti in date differenti. La loro semantica implica che $\mathcal{O}$ catturi precisamente i vincoli $S2$ delle specifiche. Dunque l'insieme di risposta di $\mathcal{G} \cup \mathcal{C} \cup \mathcal{F}_I \cup \mathcal{O}$ fornisce precisamente l'organizzazione desiderata.

Percorso Hamiltoniano

Si consideri il classico problema della teoria dei grafi conosciuto con il nome di cammino Hamiltonian. Questo problema appartiene alla classe di problemi NP-complete. La sua definizione formale è:

Definition 6.5 (HAMPATH). *Dato un grafo diretto $G = (V, E)$ e un nodo $a \in V$ di questo grafo, esiste un cammino in G con nodo iniziale a e passante una sola volta attraverso ogni nodo in V?*

Si assume che il grafo G sia specificato da un insieme di fatti sui predicati *node* e *arc*, e che il nodo iniziale a sia definito dal predicato *start*. Allora, il seguente programma **GCO** $\mathcal{P}_{hp}$ risolve il problema del cammino Hamiltoniano:

$$\left.\begin{array}{ll} inPath(X,Y) \vee outPath(X,Y) \coloneq start(X), arc(X,Y). & \\ inPath(X,Y) \vee outPath(X,Y) \coloneq reached(X), arc(X,Y). & \\ reached(X) \coloneq inPath(Y,X). & \textbf{(aux.)} \end{array}\right\} \textbf{Guess}$$

$$\left.\begin{array}{l} \coloneq inPath(X,Y), inPath(X,Y1), Y <> Y1. \\ \coloneq inPath(X,Y), inPath(X1,Y), X <> X1. \\ \coloneq node(X), \text{not } reached(X), \text{not } start(X). \end{array}\right\} \textbf{Check}$$

Le due regole disgiuntive generano un sotto-insieme S di archi come facenti parte del cammino, mentre le restanti regole controllano se S sia o meno una soluzione ammissibile per il problema. Il predicato ausiliario *reached*, utilizzato con il predicato generato *inPath*, permette di controllare la generazione e fornisce una soluzione più elegante rispetto all'utilizzo della regola

$$reached(X) \vee \neg reached(X) \coloneq node(X).$$

e la successiva richiesta di verifica se *reached* sia o meno compatibile con la generazione del predicato *inPath*. A sua volta, attraverso la seconda regola il predicato *reached* influenza la generazione induttiva di *inPath*: partendo dagli archi collegati al nodo iniziale, si estendono ripetutamente i cammini con gli archi uscenti dai nodi raggiunti. Nella parte di verifica, i primi due vincoli assicurano che l'insieme degli archi S, selezionati da *inPath*, rispetti le seguenti richieste: non esistono due archi che iniziano con lo stesso nodo, e non esistono due archi che finiscono sullo stesso nodo. Il terzo vincolo ha infine il compito di verificare che tutti i nodi del grafo siano raggiunti dagli archi dell'insieme S.

Dunque, dato un insieme di fatti $\mathcal{F}$ per *node*, *arc*, e *start*, il programma $\mathcal{P}_{hp} \cup \mathcal{F}$ produce un insieme di risposta se e solo se il corrispondente grafo ha un cammino Hamiltoniano.

La codifica del problema proposta è molto flessibile, e può essere impiegata anche per risolvere il vicino problema del calcolo del ciclo Hamiltoniano(dove il risultato deve essere un tour). Si noti che per verificare l'assenza di cicli sia sufficiente aggiungere nel programma il seguente vincolo:

$$:\!- start(Y), inPath(\_, Y).$$

Questo vincolo impone che qualsiasi insieme S sia ritornato come risposta di $\mathcal{P}_{hp} \cup \mathcal{F}$ costituisce un cammino Hamiltoniano con nodo iniziale a. Se invece è richiesto il ciclo, si deve semplicemente rimuovere il letterale "not $start(X)$" dall'ultimo vincolo del programma.

Il problema del commesso viaggiatore

Fra i problemi di ottimizzazione maggiormente studiati vi è quello del commesso viaggiatore(TSP). Questo problema può essere formalmente definito come:

Definition 6.6 (TSP). *Dato un grafo diretto pesato $G = (V, E, C)$ e un nodo $a \in V$ trovare il ciclo in G, il cui costo associato sia minimo, che abbia a come nodo iniziale e tocchi tutti i restanti nodi in V una sola volta.*

In letteratura sono presenti molti risultati sulla caratterizzazione della complessità di questo problema. Il problema di trovare una soluzione ottima è un problema intrattabile, calcolare un tour ottimale è sia NP-hard sia co-NP-hard, finanche decidere se il costo un tour ottimale è un numero pari risulta essere Δ_2^P-completo. Ciononostante è possibile codificare facilmente questo problema partendo dalla codifica proposta per il ciclo Hamiltoniano e aggiungendo le regole per esprimere la richiesta di ottimizzazione.

Si assuma che ad ogni arco del grafo G sia associato un peso, e che l'obiettivo del programma sia quello di selezionare il tour col minimo costo totale. Ancora una volta sia G specificato dai predicati *node*, *arc*, e *start*. Ma questa volta il predicato *arc* diventa ternario per poter rappresentare il peso associato all'arco. La modifica richiesta al programma per calcolare il ciclo Hamiltoniano si riduce all'aggiunta del seguente vincolo debole:

$$:\!\sim inPath(X, Y, C).\ [C : 1]$$

Questo vincolo definisce l'ordine di preferenza per gli archi presenti sul cammino. L'effetto che ha sugli insiemi di risposta è quello di selezionare quegli insiemi per i quali il costo totale degli archi contrassegnati da *inPath* sia minimo. Il programma $\mathcal{P}_{tsp}$ per risolvere il problema TSP è dunque:

$$\left.\begin{array}{l} inPath(X, Y, C) \vee outPath(X, Y, C) :\!- start(X), arc(X, Y, C). \\ inPath(X, Y, C) \vee outPath(X, Y, C) :\!- reached(X), arc(X, Y, C). \\ reached(X) :\!- inPath(Y, X, C). \qquad \textbf{(aux.)} \end{array}\right\} \textbf{Guess}$$

$$\left.\begin{array}{l} :\!- inPath(X, Y, \_), inPath(X, Y1, \_), Y <> Y1. \\ :\!- inPath(X, Y, \_), inPath(X1, Y, \_), X <> X1. \\ :\!- node(X), \text{not } reached(X). \end{array}\right\} \textbf{Check}$$

$$\left.\begin{array}{l} :\!\sim inPath(X, Y, C).\ [C : 1] \end{array}\right\} \textbf{Optimize}$$

Dato l'insieme di fatti che specifica l'istanza di ingresso, è facile mostrare l'insieme di risposta ottimale di $\mathcal{P}_{tsp} \cup \mathcal{F}$ sia in corrispondenza uno-ad-uno con i tour ottimali del grafo di partenza.

Compagnie strategiche

Un esempio di problema localizzato nel secondo livello della gerarchia polinomiale è il problema della selezione di un sotto-insieme di compagnie con determinate caratteristiche di profittabilità denominato *compagnie strategiche* [17]. Formalmente questo problema è definito come:

Definition 6.7 (STRATCOMP). *Sia data una collezione $C = \{c_1, \ldots, c_m\}$ di compagnie c_i controllate da una holding, l'insieme $G = \{g_1, \ldots, g_n\}$ di beni prodotti dalla holding, per ogni c_i si ha un sotto-insieme $G_i \subseteq G$ di beni prodotti da c_i e un insieme $O_i \subseteq C$ di compagnie che controllano c_i. O_i è denominato l'insieme controllore di c_i, e in generale una compagni potrebbe avere più di un insieme controllore associato.*

Un sotto-insieme di compagnie $C' \subseteq C$ è un insieme preservate-la-produzione se valgono le seguenti condizioni:

1. *le compagnie in C' producono tutti i beni G, ad esempio $\bigcup_{c_i \in C'} G_i = G$.*
2. *le compagnie in C' sono chiuse sotto la relazione di controllo, ovvero se $O_i \subseteq C'$ per qualche $i = 1, \ldots, m$ allora $c_i \in C'$ deve essere vera.*

Un sotto-insieme minimale C', che sia preservate-la-produzione, è detto un insieme strategico. Una compagnia $c_i \in C$ è detta strategica *se appartiene a qualche insieme strategico di C.*

In generale calcolare le compagnie strategiche è Σ_2^P-hard, riformulandolo come problema decisionale "Data una particolare compagnia c in ingresso decidere se c sia o meno strategica" diviene un problema Σ_2^P-complete. Adottando poi ulteriori specificazione, descritte in [17], che prevedono che ogni prodotto sia riferibile ad al massimo due compagnie, per ogni $g \in G \; |\{c_i \mid g \in G_i\}| \leq 2$, e che a sua volta ogni compagnia possa essere controllata da al massimo tre compagnie, $|O_i| \leq 3$ per $i = 1, \ldots, m$, si assume che ogni data istanza del problema $\mathcal{F}$ contenga i seguenti fatti:

- $company(c)$ per ogni $c \in C$,
- $prod\_by(g, c_j, c_k)$, se $\{c_i \mid g \in G_i\} = \{c_j, c_k\}$, dove c_j e c_k possono coincidere,
- $contr\_by(c_i, c_k, c_m, c_n)$, se $c_i \in C$ e $O_i = \{c_k, c_m, c_n\}$, dove c_k, c_m, e c_n non sono necessariamente distinte.

Il programma $\mathcal{P}_{strat}$, per risolvere questo problema, può essere elegantemente sviluppato con due sole regole:

$r_{s1} : strat(Y) \vee strat(Z) \text{ :- } prod\_by(X, Y, Z).$ } **Guess**

$r_{s2} : strat(W) \text{ :- } contr\_by(W, X, Y, Z), strat(X), strat(Y), strat(Z).$ } **Check**

Qui con il predicato $strat(X)$ si denotano le compagnie X ritenute strategiche. La parte generativa $\mathcal{G}$ è composta dalla regola disgiuntiva r_{s1}, mentre la parte di controllo $\mathcal{C}$ contiene la regola normale r_{s2}.

Nonostante il problema originario sia intrinsecamente difficile appartenendo alla classe Σ_2^P-hard, la sua codifica in DLV realizzata attraverso il programma $\mathcal{P}_{strat}$ è sorprendentemente succinta. Ciò accede perché il programma $\mathcal{P}_{strat}$ sfrutta la minimizzazione inerente alla semantica degli insiemi di risposta per verificare se un insieme candidato C' di compagnie, che produce tutti i beni, rispetti pure il vincolo di minimalità imposto sull'insieme di controllo.

La regola generativa r_{s1}, infatti, seleziona una delle compagnie c_1 e c_2 che sono responsabili della produzione di qualche bene g, descritto da $prod\_by(g, c_1, c_2)$. Se non esistesse nessuna informazione sul controllo delle compagnie, allora la minimalità dell'insieme di risposta sarebbe naturalmente assicurato e che quindi gli insiemi di risposta di $\mathcal{F} \cup \{r_{s1}\}$ corrisponderebbero agli insiemi strategici. Nel caso in cui l'informazione sul controllo sia disponibile, allora la regola r_{s2} deve controllare che nessuna compagnia sia venduta se è controllata da compagnie nell'insieme strategico, nel qual caso si richiede che la compagnia sia inserita in quest'ultimo insieme.

La minimalità dell'insieme strategico è garantito dalla minimalità dell'insieme di risposta. Gli insiemi di risposta di $\mathcal{P}_{strat} \cup \mathcal{F}$ sono in relazione uno-ad-uno con gli insiemi strategici per la holding. Una compagnia c è dunque strategica se e solo se il fatto $strat(c)$ appare in qualche insieme di risposta di $\mathcal{P}_{strat} \cup \mathcal{F}$.

E' importante notare qui come il vincolo r_{s2} interferisca con la regola generativa r_{s1} poiché la sua applicazione può scartare un insieme generato da r_{s1}. Si consideri, ad esempio, che r_{s1} dia origine alla seguente regola fissata r_{sg1}:

$$strat(c1) \ \texttt{v} \ strat(c2) \text{ :- } prod\_by(g, c1, c2).$$

e il fatto $prod\_by(g, c1, c2)$ sia presente in $\mathcal{F}$. Si supponga ora che la regola sia soddisfatta in r_{s1} rendendo $strat(c1)$ vero. Se, tuttavia, un istanza della regola r_{s2} una volta applicata deriva $strat(c2)$, allora la precedente applicazione della regola r_{sg1} per derivare $strat(c1)$ è invalidata. Ciò accade poiché la minimalità degli insiemi di risposta impongono che $strat(c1)$ non possa essere derivato dalla regola r_{sg1}, se un altro atomo nella sua testa è vero.

Per le considerazioni sulla complessità discusse nelle sezioni precedenti, questo modo di procedere è necessario per il risolvere il problema STRATCOMP senza dover ricorrere a regole disgiuntive nella parte di verifica, poiché decidere se una particolare compagnia sia o meno strategica è Σ_2^P-complete. Una rappresentazione alternativa che esprime gli insiemi strategici nel generico paradigma **GCO** è presentata in [37].

Usando delle tecniche come quelle descritte in [38], il programma $\mathcal{P}_{strat}$ può essere esteso così da rimuovere i due vincoli definiti prima sul massimo numero di compagnie producenti un bene, e sul numero di compagnie che possono controllare una particolare compagnia.

Compagnie strategiche preferite

E' possibile estendere il problema discusso nella precedente sezione così da attribuire alle compagnie un ordinamento di preferenza. Tale preferenza potrebbe essere espressa, nel caso di insieme strategici multipli, a favore di quegli insiemi le cui compagnie non sono in procinto di essere vendute e sussiste una forte volontà nel mantenerle all'interno della holding. In presenza di tali preferenze, il problema aumenta leggermente la sua complessità passando da Σ_2^P a Δ_3^P.

Si assuma che le preferenze siano espresse attraverso attraverso un unico predicato $avoid(c_{sell}, c_{keep}, pr)$, che definisce la priorità di vendere c_{sell} mantenendo c_{keep} al valore pr. Sulla informazioni di questo predicato è possibile aggiungere alla precedente codifica del problema il seguente vincolo debole:

$$:\sim avoid(Sell, Keep, Priority), \text{not } strat(Sell), strat(Keep). \ [: Priority]$$

6.4 Sommario e ulteriori riferimenti

La programmazione logico disgiuntiva è ormai un formalismo molto apprezzato all'interno della comunità dell'inteligenza artificiale, soprattutto per la sua capacità espressiva. Fra i lavori principali su DLP vi sono [2, 93, 53, 92, 108, 1, 84]. Particolare interesse ha rivestito nel tempo la sua semantica come testimoniato dai lavori [12, 41, 107, 123, 144], solo per citarne alcuni.

Oggigiorno la più accettata definizione di semantica è riferita alla proposta di Gelfond e Lifschitz presentata in [53]. I risultati principali sull'espressività possono essere trovati nei lavori [97, 39, 40, 30].

Dopo molti anni di ricerca il sistema DLV fu presentato in [84], e rappresenta ancora oggi una delle implementazioni più efficienti di motore d'inferenza basato su DLP.

Una recente analisi dello stato dell'arte delle tecnologie ASP è fornita nel lavoro [18] che descrive in dettaglio tutti gli aspetti relativi all'evento denominato *ASP Competition*. Quest'ultimo ha come principale obiettivo la valutazione dei progressi delle più recenti tecnologie, valutate su diversi ambiti applicativi, e la standardizzazione del linguaggio "ASP-Core-2".

7

Interrogazioni in Linguaggio Naturale ad una Base di Conoscenza

Questo capitolo presenta il principale contributo della tesi: le tecniche per l'interpretazione di una interrogazione in linguaggio naturale Q_{NL} ad una base di conoscenza KB che si assume rappresentata da un programma Datalog. L'interpretazione descritta sfrutta anche alcune risorse esterne quali BabelNet, ConceptNet e FrameNet, descritte rispettivamente nelle sezioni 4.4, 4.2.3 e 4.2.2, per gestire in modo appropriato la similarità, anche semantica, tra i concetti menzionati nell'interrogazione in input e quelli presenti nella base di conoscenza.

Come già descritto nell'introduzione, le assunzioni principali da considerare sono:

- si intende rispondere solo alle interrogazioni che sono perfettamente "coperte" dalla base di conoscenza, pertanto vale l'assunzione di mondo chiuso che, in qualche modo, possiamo considerare non solo per i dati ma anche per i concetti coinvolti (se un concetto menzionato nell'interrogazione Q_{NL} non è presente in KB allora non possiamo esprimerci in merito e lo notifichiamo all'utente);
- nella progettazione di una base di conoscenza Datalog sia i simboli di predicato sia le eventuali costanti presenti nell'IDB sono identificati da termini intellegibili, cioè in grado di rendere immediatamente evidente il proprio significato ed il proprio contributo nell'ambito delle regole in cui occorrono;
- predicati e regole dell'IDB della base di conoscenza rappresentano in modo compatto una serie limitata di possibili interrogazioni a KB. Tipicamente tali predicati non sono molti, rispetto alla mole dei dati presenti nella parte estensionale (EDB), e possiamo assumere che possano essere annotati, in modo semi-automatico, per legare in modo più preciso i termini ed i loro ruoli nell'IDB della base di conoscenza con i concetti presenti nelle risorse semantiche a disposizione. In altri termini, sfruttiamo il fatto che i dati sono strutturati e, in termini di basi di dati, stiamo annotando so-

lo lo schema (ed eventualmente i vincoli) della base di dati per ottenere automaticamente la caratterizzazione di tutto ciò che vi è memorizzato.

Nelle prossime sezioni descriveremo il processo di interrogazione della KB, che consta delle seguenti fasi:

1. annotazione della base di conoscenza;
2. elaborazione dell'interrogazione Q_{NL}:
   a) analisi sintattica;
   b) determinazione del tipo di interrogazione;
   c) semplificazione ed eventuale ristrutturazione delle dipendenze;
   d) generazione programma Datalog intermedio LIP;
3. costruzione del programma Datalog Q_D che rappresenta l'interrogazione Datalog "eseguibile".

Si noti che la prima fase è effettuata "offline", cioè è indipendente dalla particolare interrogazione che viene effettuata. Solo le operazioni 2 e 3 hanno quindi una caratteristica interattiva e devono essere implementate con particolare efficienza. Per questo motivo, l'approccio che si propone concentra moltissimi sforzi computazionali nella fase 1, che può essere vista come una sorta di *compilazione* delle principali interrogazioni (sempre in termini di struttura e non di dati restituiti) che possono essere effettuate alla base di conoscenza KB.

Un aspetto rilevante sul quale si è posta particolare attenzione è l'efficacia del framework proposto, che si sostanzia di usabilità e flessibilità, oltre ovviamente agli aspetti di efficienza dell'esecuzione e di correttezza e completezza delle risposte. Per questo motivo le annotazioni alla base di conoscenza sono sostanzialmente effettuate in linguaggio naturale e poi analizzate ed elaborate in automatico. Il risultato è però costituito da moduli Datalog che possono/devono essere supervisionati da chi gestisce la base di conoscenza e, soprattutto, possono essere ulteriormente modificati e arricchiti secondo le esigenze, così da catturare ulteriori aspetti semantici che non dovessero essere stati colti dagli algoritmi proposti con le conoscenze semantiche a disposizione.

7.1 Analisi Sintattica

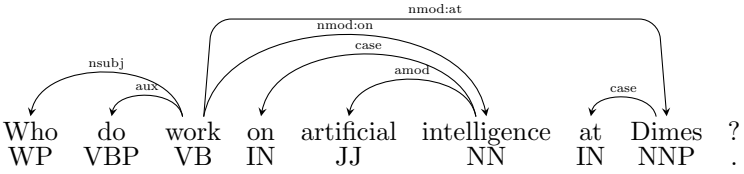
Preliminarmente descriviamo le informazioni ottenute mediante l'analisi sintattica del linguaggio naturale, uno strumento importante che usiamo sia per l'elaborazione dell'interrogazione in input sia per le annotazioni della base di conoscenza.

Una frase $\bar{w}$ in linguaggio naturale è rappresentata da una sequenza di stringhe alfanumeriche $w_1, \ldots, w_n$, che chiamiamo *termini*. L'analisi sintattica di $\bar{w}$ prevede una serie di operazioni che comprendono la segmentazione,

la normalizzazione[1], l'analisi delle parti del discorso ed infine l'analisi delle dipendenze. Poiché molto lavoro è stato fatto su questo fronte, è preferibile impiegare i framework oggi disponibili che, almeno per la lingua inglese, funzionano molto bene. Gli strumenti messi a disposizione dal gruppo di linguistica computazionale dell'università di Stanford[2], ad esempio, garantiscono prestazioni di affidabilità molto alta su tutti gli aspetti qui richiesti. In particolare l'analisi sintattica riduce i termini $w_1, \ldots, w_n$ nella loro forma canonica, li annota con le opportune parti del discorso, ed esegue l'elaborazione delle relazioni di dipendenza utilizzando il parser proposto da Chen e Manning e descritto nella sezione 1.6.

Per quello che interessa il presente lavoro, l'output dell'analisi sintattica di una frase $\bar{w}$, che chiamiamo annotazione sintattica o grafo delle dipendenze, è un grafo orientato $G_{\bar{w}} = (N, D, pos, dep)$ etichettato su nodi e archi tale che: l'insieme dei nodi N è l'insieme dei termini di $\bar{w}$; per ogni $w_i \in N$, l'etichetta $pos(w_i)$ identifica la parte del discorso associata al termine w_i; per ogni arco $(w_i, w_j) \in D$, l'etichetta $dep(w_i, w_j)$ esprime una delle relazioni definite nella grammatica delle dipendenze universali utilizzata nel framework di Stanford (e riportata in appendice, per completezza).

Example 7.1. Si consideri la domanda nq_1:= *"Who do work on artificial intelligence at Dimes ?"*. L'analisi produce una annotazione sintattica che riportiamo graficamente di seguito (l'etichetta *pos* di ogni nodo è riportata sotto il nodo/termine):



In questo esempio le relazioni importanti al fine dell'approccio proposto sono un sottoinsieme delle relazione annotate dal parser e mostrate sopra. Più in dettaglio la relazione etichettata con *nsubj*, che identifica il soggetto, è assunta fondamentale ai fini dell'analisi semantica. Anche *amod* che, come annotazione di una forma aggettiva, specifica ulteriormente l'entità da cui dipende sarà presente dell'analisi. Saranno invece escluse dall'analisi semantica relazioni come *aux* e *case*. La prima esplicita il legame fra verbo principale e un verbo ausiliario richiesto per esprimere un particolare costrutto. Sotto l'assunzione che le espressioni in ingresso siano tutte delle interrogazioni e non essendo presente nel sistema nessun meccanismo per la gestione cronologica degli eventi tale informazione non fornisce nessun contributo all'analisi

[1] Per normalizzazione si intende l'estrazione estrazione del lemma o della forma canonica di un termine: ad esempio la forma plurale è ridotta alla sua forma singolare (*articles* diviene *article*), mentre le forme verbali sono ricondotte alla radice del verbo (*going* diviene *go*).

[2] http://nlp.stanford.edu/software/

semantica. Lo stesso vale per la seconda, che pur legando un'espressione nominale alla preposizione che la introduce, risulta essere ridondante per via dell'arricchimento di altre relazioni.

Infine la relazione *nmod*, la cui modificazione nominale riceve un'ulteriore specificazione attraverso l'aggiunta della preposizione, è utilizzata per definire diversi percorsi di interpretazione durante l'analisi. Una discussione più approfondita su questi argomenti sarà fornita nelle sezioni successive.

7.2 Modello per l'annotazione dei predicati

In questa sezione presentiamo il modello usato per annotare la base di conoscenza KB, in particolare per annotarne i predicati.

Remark 7.2 (Assunzioni). Per semplicità assumiamo senza perdita di generalità che ogni simbolo di predicato sia usato in KB con una sola arità, sicché il predicato è univocamente rappresentato dal proprio nome. Assumiamo inoltre, per semplicità, che ogni entità di interesse sia identificata univocamente da un singolo argomento, in altri termini assumiamo che le chiavi delle relazioni usate nell'EDB siano dei singoletti.

Come già illustrato nell'introduzione al capitolo, l'approccio qui descritto sfrutta il fatto che, tipicamente, la scelta dei termini che occorrono nella base di conoscenza è fortemente legata alla semantica di ciò che viene predicato, cioè del significato delle estensioni di ogni predicato negli answer set. Alcuni aspetti comunicativi del significato di una base di conoscenza sono tuttavia implicitamente ricondotti all'interpretazione umana e, pertanto, spesso preclusi ai sistemi che dovessero interagire con essa in modo automatico e non supervisionato da un umano.

Per superare questo problema, nel presente lavoro assumiamo che il progettista della base di conoscenza (o comunque colui che è incaricato di interfacciarla col resto del mondo) provveda ad annotarne ciascun predicato con semplici informazioni di tipo semantico e con qualche indicazione sul ruolo svolto da ogni argomento nell'ambito del predicato stesso. Si noti infatti che non si sta assumendo alcuna limitazione sull'arità dei predicati in KB, sicché ogni predicato può rappresentare facilmente molteplici relazioni tra i dati associati ai suoi argomenti.

Ciò rappresenta ovviamente un problema significativo, ma anche una peculiarità di questo lavoro rispetto a precedenti esperienze in letteratura che si sono invece concentrate su modelli ristretti a relazioni binarie o a triple, cioè a schema fissato a priori per tutta la base di dati (e di conseguenza privo di una propria rilevanza semantica) [7, 119, 120, 129]. Tra i lavori correlati sotto questo aspetto ricordiamo anche gli approcci descritti in letteratura per arricchire ontologie o, più in generale, basi di conoscenza con informazioni ausiliarie. Ad esempio il modello lessicale per ontologie, *LeMOn*[3], nasce appunto

[3] http://lemon-model.net/

col l'intento di fornire le possibili lessicalizzazioni per i concetti modellati da un'ontologia. Secondo le specifiche di questo modello, per ogni elemento dell'ontologia è prevista un'entità lessicale associata e corredata da riferimenti ad informazioni morfologiche e sintattiche. Altri approcci più recenti, invece, sfruttano tecniche di apprendimento automatico per arricchire le entità della base di conoscenza con vettori di parole. Indipendentemente dalla tipologia di entità adottata, lo scopo di tali modelli rimane comunque lo stesso: facilitare l'allineamento di espressioni linguistiche a concetti definiti formalmente.

7.2.1 Il Modello PSA

Il modello di annotazione per i predicati che qui si propone, denominato *PSA*, generalizza i precedenti approcci in quanto è in grado di gestire predicati di arità arbitraria e considera diverse tipologie di informazioni aggiuntive per permettere ulteriori forme di inferenza. Esso prevede informazioni lessicali, sintattiche e riferite al senso-comune della predicazione e delle entità in essa coinvolte.

Sul piano lessicale, PSA prevede che il simbolo di predicato ed i suoi argomenti siano descritti da insiemi di espressioni linguistiche (anche multiparole), relativi vettori di parola e possibilmente da riferimenti ai synset di BabelNet o ad altra risorsa similare. Ogni insieme è caratterizzato da elementi per i quali sussiste una relazione di similarità semantica circoscritta alla situazione descritta dal predicato.

Per esplicitare i ruoli svolti dai diversi argomenti rispetto al predicato in cui occorrono, sono inoltre fornite alcune descrizioni espresse in linguaggio naturale, ad esempio "*Il ricercatore X afferisce al dipartimento Y*". Tali descrizioni, in cui sono esplicitate le parole usate nei predicati ed i riferimenti agli argomenti del predicato stesso, sono accompagnate dalle annotazioni sintattiche descritte nella precedente sezione.

Si utilizzano infine le relazioni presenti su ulteriori risorse esterne quali ConceptNet o BabelNet per rappresentare conoscenza di senso-comune relativa a quel predicato. Tecnicamente si tratta di estendere KB con nuova conoscenza, rappresentata da sottoprogrammi Datalog che chiamiamo moduli, che può essere stata considerata implicita (per gli umani) in KB ma che, se esplicitata, può rendere più efficace il sistema per query in linguaggio naturale (in cui l'utente potrebbe appunto sottintendere tale conoscenza). Si pensi ad esempio a classiche relazioni IS-A come quelle che esprimono il fatto che un professore è una persona, un ricercatore è una persona, un tecnico è una persona, etc. Per rispondere ad una query del tipo "le persone che afferiscono al DIMES" sarebbe ovviamente importante poter disporre di tale informazione. Altri esempi possono riguardare il concetto di (personale) dipendente e così via. È importante notare che, grazie alla flessibilità di Datalog, ulteriori regole che si ritengano utili per le interrogazioni ma che non sono coperte dalle risorse semantiche a disposizione possono essere facilmente aggiunte ai moduli in qualunque momento.

Sia P l'insieme dei predicati che occorrono in KB (come detto identificati dal proprio simbolo), N_C l'insieme delle costanti che occorrono nelle regole IDB di KB e $V = P \cup N_C$. È inoltre dato un lessico L (fornito di espressioni multi-parola), un insieme WE di vettori di parole e una funzione $f_{we} : L \to 2^{WE}$ che associa ad elemento $l \in L$ un sottoinsieme di WE. Denotiamo inoltre con BN l'insieme degli identificativi dei synset di BabelNet e con FUN l'insieme dei ruoli funzionali, tratti da ConceptNet e illustrati in Figura 7.2.1, che useremo per le annotazioni.

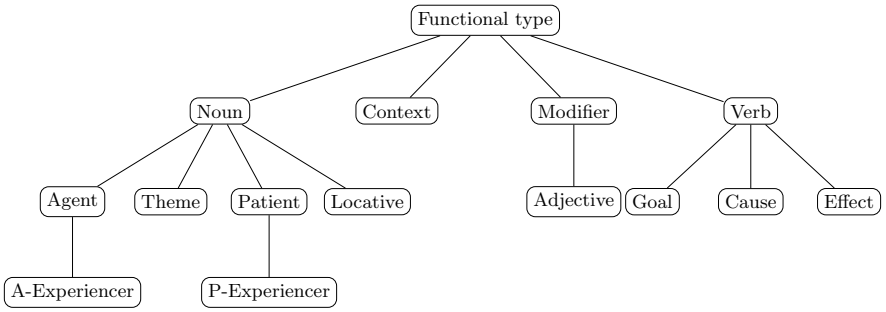


Figura 7.1. Tassonomia dei ruoli in *FUN*.

Definizione 7.2.1 L'annotazione PSA_p di un predicato $p \in P$ avente arità r è una tripla $\langle Lex, Syn, Mod \rangle$, in cui

Lex è una sequenza triple della forma $\langle l_i, we_i, bn_i \rangle$, dove $l_i \in L$, $we_i \in f_{we}(l_i)$ e $bn_i \in BN$, tale che $|Lex| = r + 1$ (abbiamo cioè un elemento per il predicato ed uno per ciascuno dei suoi argomenti) ed ogni elemento di L può occorrere al più in una tripla;

Syn è una sequenza di coppie della forma $(\bar{w}, arg)$, in cui

- $\bar{w}$ è una frase in linguaggio naturale che illustra il predicato p;
- arg è una funzione che associa ad ogni indice $0 \leq j \leq |Lex|$ (cioè al nome del predicato o ad uno dei suoi r argomenti) un sottoinsieme di termini $arg(j)$ che occorrono in $\bar{w}$; si noti che $arg(j)$ può essere vuoto se l'elemento (di indice) j non è rilevante per la descrizione $\bar{w}$;

Mod è un insieme di regole Datalog, tipicamente semplici fatti derivati automaticamente dalle descrizioni e dalle risorse semantiche a disposizione, che formalizza i ruoli degli elementi rilevanti per il predicato p e come essi siano fra di loro collegati attraverso relazioni di senso comune. I simboli di constante che occorrono nel programma possono essere riferimenti ad

- *inner concepts* (*i_conc,*), cioè a termini che occorrono nel predicato, ovvero negli altri elementi dell'annotazione; oppure ad

- *external concepts* (*e_conc,*), cioè riferimenti a concetti esterni alla predicazione ma legati ad essi attraverso relazioni di senso comune presenti nelle risorse semantiche a disposizione.

I simboli di predicato di *Mod* sono l'insieme di relazioni usate nel framework ConceptNet, i predicati unari riferiti alle tipologie delle costanti {*i_conc,e_conc*} ed il predicato binario *hasFunctType*.

In *Lex* la prima posizione del vettore è occupata dalla costante o dal simbolo di predicato. In quest'ultimo caso, il vettore ha dimensione uguale all'arità del predicato più uno così da avere nella posizione di indice i l'annotazione lessicale per l'i-esimo argomento di p. Gli elementi del vettore sono liste di realizzazioni lessicali accompagnate da una loro rappresentazione vettoriale e un riferimento al synset di BabelNet. Come sarà descritto più in dettaglio nel seguito, questi ultimi due elementi saranno utilizzati nella fase di allineamento degli elementi della domanda agli elementi formali della base di conoscenza.

Syn fornisce esempi illustrativi delle realizzazioni linguistiche associabili ad una costante o ad un predicato e ai suoi argomenti. Estendere gli elementi della base di conoscenza con evidenze della loro manifestazione nelle espressioni naturali agevola la risoluzione dei loro ruoli tematici.

L'insieme delle definizioni è rivolto a catturare (anche se in modo ristretto) la variabilità permessa dal linguaggio naturale nell'esprimere la situazione descritta dal predicato. Tali descrizioni, benché non formalmente vincolate in forma o contenuto, vanno espresse in una struttura sintattica il più possibile semplice, evitando costrutti sintattici che possano portare ad ambiguità strutturale come sequenze di clausole relative, quantificatori innestati, o espressioni ellittiche. Seguire questo principio ha il duplice vantaggio di rendere più agevole sia la comprensione della definizione da parte di un utilizzatore terzo della base di conoscenza, sia il suo utilizzo per scopi d'inferenza.

Infine l'insieme di regole Mod rappresenta una formalizzazione delle situazioni espresse dalle entità di KB secondo il modello concettuale proposto da ConceptNet, esplicitando formalmente i legami fra le entità coinvolte. Definisce inoltre vincoli di natura ontologica e funzionale su tali entità.

I vincoli ontologici sono in parte già definiti attraverso l'associazione, descritta in *Lex*, tra il predicato ed i suoi argomenti e le informazioni della rete BabelNet. La metodologia adottata evita una descrizione accurata dell'argomento in termini di assiomi logici; invece essa si limita a definire la tipologia dell'argomento in termini di synset di BabelNet e accettare come argomento compatibile qualsiasi iponimo di tale concetto.

I vincoli funzionali, infine, sono definiti a partire dai ruoli semantici definiti nella tassonomia in Figura 7.2.1, ed assegnati agli elementi dell'annotazione mediante il predicato *hasFunctType*.

L'idea alla base dell'utilizzo delle relazioni di senso comune per descrivere l'interazione fra gli elementi della predicazione risiede nella possibilità di rappresentare attraverso tali relazioni la funzione che un'entità gioca all'interno dello scenario descritto dal predicato.

La lista delle regole impiegate per l'assegnamento dei ruoli funzionali a partire dalle informazioni presenti in Syn (e dalle associate annotazioni sintattiche) è mostrata nella tabella seguente:

Tabella 7.1. Regole di assegnamento funzionale

Tipo	Regola
Actor definer	*hasFunctType(X,agent) :- capableOf(X,Y).*
Object definer	*hasFunctType(Y,patient) :- capableOf(X,Y).*
Location definer	*hasFunctType(Y,locative) :- atLocation(X,Y).*

Example 7.3. In tabella 7.2 si riporta una possibile annotazione per l'atomo $research(X, Y, Z) \in KB$ utilizzato per descrivere un'attività di ricerca svolta da X su un argomento Y all'interno di un dipartimento Z.

Tabella 7.2. Annotazione PSA per il predicato *research(X,Y,Z)*

Lex	$\langle''research'', v_{research},'' bn : 00095823v''\rangle$, $\langle''researcher'', v_{researcher},'' bn : 00047356n''\rangle$, $\langle''topic', v_{topic},'' bn : 00074900n''\rangle$ $\langle''department'', v_{dept},'' bn : 00026308n''\rangle$,
Syn	*"A scientist researches on a topic at a department." :* $\{(''scientist'', 1), (''researches\ on'', 0), (''topic'', 2), (''department'', 3)\}$;
Mod	$isA(research_{arg0}, bn00095823v)$. $isA(research_{arg1}, bn00047356n)$. $isA(research_{arg2}, bn00074900n)$. $isA(research_{arg3}, bn00026308n)$. $capableOf(research_{arg1}, research_{arg0})$. $hasProperty(research_{arg0}, research_{arg2})$. $relatedTo(research_{arg0}, research_{arg2})$. $atLocation(research_{arg0}, research_{arg3})$.

Osservazione 1 *Il modello d'annotazione PSA segue le intuizioni alla base della risorsa lessicale FrameNet nella misura in cui essa abbraccia il principio di evocazione e circoscrizione semantica definita dalla semantica dei frame. E' infatti immediato considerare la situazione espressa attraverso un predicato come un frame in miniatura, all'interno del quale si realizzano dei particolari ruoli semantici.*

Le unità lessicali, che in FrameNet hanno il compito di evocare la situazione definita dal frame, sono qui le realizzazioni lessicali presenti in Lex che emulano l'azione evocativa abilitando la costante o il predicato collegato.

Si osserva comunque che gli scopi delle due annotazioni sono differenti. Lo scopo di FrameNet è quello di fornire una collezione di testi annotati secondo la semantica di frame. Nel fare ciò FrameNet fornisce concrete realizzazioni delle situazioni reali descritte dai frame, evidenzia la presenza di determinati elementi all'interno della situazione e, infine, seleziona un insieme di parole accomunate dalla loro capacità evocativa verso una stessa situazione. Il modello PSA, invece, ha come principale obiettivo la costruzione di legami fra elementi formali della base di conoscenza e gli elementi e i costrutti del linguaggio naturale, estendendo i tradizionali approcci per lessicalizzare ontologie o, più in generale, sistemi formali.

Tali annotazioni sono tipicamente definite da chi si occupa di modellare la conoscenza, anche in linguaggio naturale, eventualmente in modo interattivo.

Un esempio più completo di annotazione è mostrato in Appendice. Si tratta dell'annotazione della base di conoscenza (semplificata) relativa alle attività di ricerca dell'Università della Calabria, usata per le sperimentazioni preliminari di questo lavoro. Le informazioni contenute nella base di dati estensionale (non riportata nell'appendice) sono quelle reali estratte dalla scheda annuale per la ricerca (SUA-RD) predisposta dal Ministero dell'Istruzione, dell'Università e della Ricerca.

7.3 Elaborazione della Domanda

In questa fase si procede all'analisi di una domanda in linguaggio naturale $nq = w_1 \dots w_n$.

Annotazione sintattica

In primo luogo si procede all'analisi sintattica, usando gli strumenti già descritti in Sezione 7.1. Sia $G_{nq} = (N, D, pos, dep)$ l'annotazione sintattica espressa dal grafo delle dipendenze associato ad *nq*. Come *running example*, consideriamo ancora la domanda *"Who do work on artificial intelligence at Dimes ?"*, la cui analisi sintattica produce l'annotazione descritta nell'Esempio 7.1.

Ottenuta tale annotazione, vengono individuati alcuni elementi notevoli:

- tipologia di domanda, *Qtype(nq)*;
- costrutti associati a predicati *built-in*, ad esempio frammenti di frase come *"how many"*, identificati mediante una annotazione chiamata *ane* (per elemento naturale annotato);
- output desiderato *Out(nq)*.

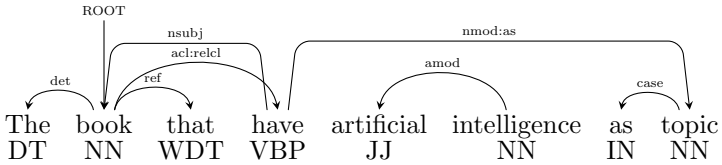
Nel running example la domanda è della tipologia *who* e non vi sono calcoli legati a conteggio o a confronti numerici. In questo lavoro assumiamo per

semplicità che l'interrogazione riguardi una singola richiesta, quindi *Qtype(nq)* è un mapping su un singolo nodo in N.

Out(nq) in questo caso si riferisce allo stesso nodo "Who" e sarà compito delle fasi seguenti definire opportunamente una predicato di output per restituire le informazioni desiderate. Questo può essere visto come un sottoproblema a parte, infatti tipicamente nei predicati della base di conoscenza le entità rilevanti sono identificate con dei codici (usati come chiavi o come chiavi esterne nei predicati collegati), mentre l'utente si aspetta in output delle informazioni più descrittive. Come vedremo l'interpretazione dell'interrogazione si occuperà anche di identificare alcune informazioni *di default* da restituire in output. Nel nostro esempio, probabilmente le informazioni saranno il nome ed il cognome della persona individuata, ma potrebbe anche essere restituito il settore scientifico-disciplinare, in base agli attributi trovati nella relazione con le informazioni anagrafiche principali.

Clausole relative e domande *WP* e *WPB*

Le specifiche descritte dalle dipendenze universali prevedono che il riferimento del pronome sia collegato con l'elemento della frase che lo governa attraverso la relazione *acl:relcl*. Questo tipo di analisi quindi facilita i riferimenti necessari per la costruzione della forma logica. Si consideri il seguente esempio:



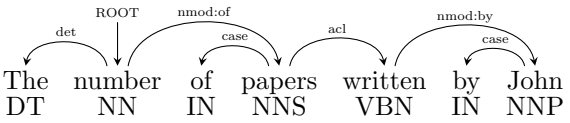
Qui il riferimento di *that* è facilmente identificabile con il nodo *book*. Una sua gestione immediata prevederebbe la gestione di una denotazione linguistica rappresentata dall'elemento governante, cioè il nodo associato a *book*. Tale informazione risulterebbe tuttavia ridondante, poiché la rimozione del nodo *that* non modifica eccessivamente la topologia del grafo (il nodo *that* non possiede dipendenti). Sulla base di questa considerazione si procede alla rimozione dei nodi contrassegnati dalle parti del discorso come WP, WDT, etc.

La presenza di parti del discorso quali WP, WDT, etc., sono indicativi della presenza di una clausola relativa. In questo contesto si assume che queste ultime siano tutte di tipo restrittivo, e non descrittivo. Tutte le clausole impongono quindi dei vincoli alle entità riferite. Come sarà mostrato nella prossima sezione tali vincoli saranno esplicitati in fase di costruzione della forma logica riferita alla domanda. E' importante sottolineare, tuttavia, come l'analisi delle clausole relative complesse può portare a strane annotazioni da parte degli strumenti linguistici. Attualmente, poiché al di fuori degli scopi di questo lavoro, non è impiegata nessuna tecnica per risolvere queste imperfezioni.

Infine si noti come il corretto trattamento delle parole *Wh-* richieda una valutazione delle loro posizioni all'interno dell'espressione linguistica. Nel caso più semplice esse appaiono solamente in testa alle domande. Il loro riferimento al predicato non-ancora-risolto definisce la variabile d'interesse nella domanda. Tuttavia occorre evidenziare come questi elementi variabili, diversamente dal caso delle clausole relative, siano privi di una denotazione linguistica. Come sarà discusso nella prossima sezione la loro denotazione, o caratterizzazione semantica, sarà indotta dal particolare argomento assegnato loro durante l'analisi basata su PSA.

Richieste di conteggio e comparazione

Espressioni comuni come "*how many*" o "*number of*" sottendono l'idea di conteggio. Al fine di esplicitare questa richiesta (implicita nella domanda) si individuano all'interno della domande le più comuni realizzazioni linguistiche e le si caratterizzano associando loro un particolare tipo nella annotazione del nodo. Si consideri, a titolo esemplificativo la seguente domanda:



La ricerca di costrutti di conteggio o comparazione identifica l'espressione "*number of*" come evidenza di un'operazione di conteggio sul suo diretto dipendente "*papers*". Per esplicitare tale richiesta, all'ane_{number} è assegnato il tipo *COUNT_D*. Durante la costruzione dell'interpretazione, la presenza di questo tipo assegnerà la funzione *COUNT* al predicato rappresentante il dipendente collegato attraverso la dipendenza *nmod:of*, in questo caso" *papers*".

In modo del tutto analogo l'espressione "*how many*" riceve il tipo *COUNT_G*. Questo tipo indicherà al sistema che la definizione della funzione di conteggio avverrà sul predicato governante il nodo *many*. Anche se all'apparenza questo processo può sembrare simile, le conseguenze dell'assegnazione della funzione di conteggio ai nodi superiori richiede un trattamento leggermente più elaborato rispetto al caso precedentemente discusso, per il quale si rimanda alla sezione del trattamento dell'interpretazione in presenza dei predicati speciali.

Catturare l'intenzione di confronto richiede un'analisi più ampia. Le comparazioni nella loro forma canonica sono introdotte da avverbi comparativi come *more*, *less*, o *as*, presentano poi un termine sul quale eseguire il confronto, e si concludono con una frase preposizionale[4] introdotta da *than*. Normalmente i modificatori dei nomi ricevono come relazione di dipendenza *amod*,

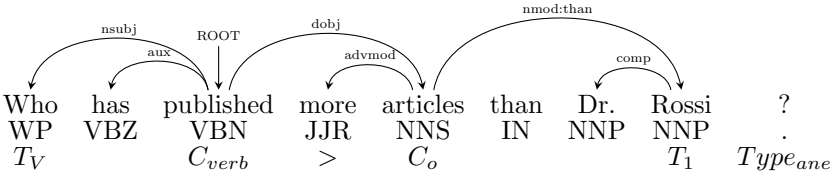
[4] In questo contesto non si considera la comparazione che utilizza una clausola invece di una frase preposizionale come "On the one hand, AI researchers are high demanded; on the other hand, they have very specific profiles which prevents their careers. ".

ma nei costrutti comparativi essi ricevono l'etichetta *advmod*. La frase preposizionale dipende comunque dal nome o dall'aggettivo sul quale eseguire la comparazione.

Le diverse tipologie di comparazione permesse dal sistema sono legate agli strumenti messi a disposizione dal linguaggio della base di conoscenza. Sulla base degli operatori di confronto offerti da DLV, $\{>, >, \leq, \geq, =, != \}$, si definiscono i corrispettivi tipi da associare alle annotazioni *ane*. La scelta di quale operatore utilizzare è realizzata attraverso liste di termini associate ad ogni tipologia, ad esempio l'espressione "*at most*" riceve come tipo $\leq$, "*less*" $<$ e così via.

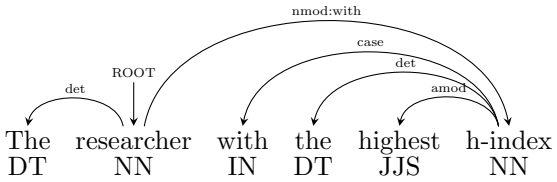
La scelta dei termini di paragone e dell'oggetto della comparazione si fonda invece su alcune considerazioni sintattiche. I termini da mettere in relazione sono associati al soggetto del verbo dell'espressione linguistica, alla quale è associato il tipo C_{verb}, che ha come oggetto l'elemento modificato nominalmente da *than* e al nodo dipendente di quest'ultima relazione. I tipi possibili da associare per questi termini sono $\{T_1, T_2, T_V\}$, questi possono essere usati in due prefissate configurazioni $\langle T_V, T_1 \rangle$ o $\langle T_1, T_2 \rangle$. La configurazione che prevede T_1 e T_2 è riferita ad espressioni di confronto i cui termini di confronto sono definiti nell'espressione linguistica, ed è riconducibile a domande *SI/NO* come "*Has A done less something than B*". Il tipo T_V è invece impiegato in presenza di un termine non specificato e quindi variabile. Infine il nodo governante l'elemento JJR riceve come tipo C_o che definisce quale sia l'oggetto sul quale operare il confronto.

Si consideri la seguente domanda posta al sistema:



Dall'analisi sintattica il sistema individuerà la presenza di una comparazione sfruttando l'annotazione JJR e assegnerà all'ane_{more} il tipo ">". Gli elementi da confrontare ricevono rispettivamente T_V e T_1, mentre *articles* riceve come tipo C_o. Per questo tipo di comparazioni, l'interpretazione sottintende che la prima parte dell'interrogazione con verbo ed oggetto (in questo caso *published articles*) è da ripetere con l'agente che segue "than" come soggetto. Nell'esempio vuol dire considerare la sottofrase implicita: "*... than Dr. Rossi published articles.*"

Infine il caso di espressioni con superlativo è gestita nella sua forma più semplice: un solo superlativo che modifica un nome. Poiché questo costrutto non riceve nessuna particolare annotazione, ad eccezione della parte del discorso JJS, l'individuazione di questo costrutto avviene esclusivamente in sua presenza. Nella domanda seguente:



Le funzioni speciali utilizzate qui sono le funzioni *max* e *min*.

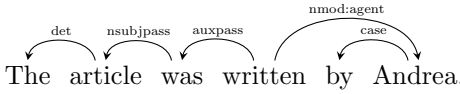
Per concludere questa veloce discussione sull'analisi comparativa si sottolinea come l'oggetto della comparazione sia assunto numerico o comunque enumerabile in qualche modo. L'atto di comparazione è per sua natura quantitativo e i termini da comparare sono selezionati basandosi su considerazioni sintattiche riferite alle più frequenti espressioni linguistiche riferite all'idea di confronto.

Inversione della forma passiva

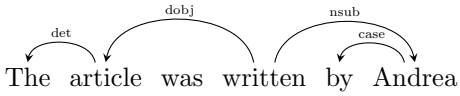
In alcune situazioni è conveniente tradurre una forma passiva in una corrispondente forma attiva, soprattutto per avere un unico modo di rappresentare il verso di relazione fra soggetto e oggetto e quindi permettere un trattamento più uniforme durante l'analisi.

Le regole d'inversione possono essere facilmente applicate in presenza delle categorie sintattiche che presentano come suffisso *-pass*. In questi casi, infatti, è facile identificare, se presenti, il soggetto e l'agente dell'espressione passiva.

Individuati gli elementi della forma passiva, tutte le dipendenze terminanti con *-pass* sono rimosse e sostituite con le loro forme attive[5]. Per la frase "The article was written by Andrea", ad esempio, si ha la seguente annotazione:



che dopo l'applicazione delle regole diviene:



Eliminazione dipendenze superflue

L'annotazione sintattica prodotta dal parser è ricca d'informazione, non completamente utilizzata nell'attuale processo di interpretazione. Ad esempio la relazione *aux*, che lega un verbo con il suo ausiliario, in questo contesto risulta inutile poiché, nella proposta corrente, non è prevista una gestione delle

[5] L'applicazione delle regole non cambia la struttura della frase, le modifiche coinvolgono solo un sottoinsieme delle annotazioni sintattiche.

diverse forme verbali[6]. Altre dipendenze risultano invece ridondanti a causa delle annotazioni arricchite di UD, come nel caso della preposizione *in front of*. In questo esempio infatti si possono eliminare le dipendenze, *case* e *mwe*, poiché la preposizione è già catturata nella sua interezza dalla dipendenza *nmod:in_front_of*. Come si vedrà nel seguito ridurre il numero di dipendenti semplifica la fase composizionale della costruzione dell'interpretazione.

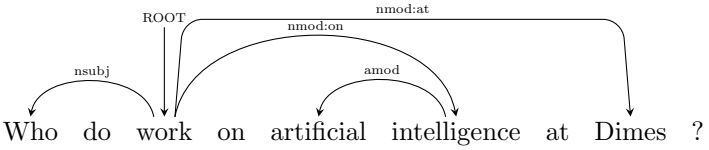
7.4 Costruzione dell'interpretazione

Conclusa la fase riferita all'analisi linguistica, si inizia la fase di interpretazione attraverso la costruzione di un programma logico che descrive l'interazione fra le diverse componenti della domanda. La capacità generativa del linguaggio naturale rende spesso difficile sia un riferimento diretto fra gli elementi della domanda e le entità della base di conoscenza, sia la definizione della loro interazione. Tuttavia la disponibilità delle annotazioni e la flessibilità del linguaggio Datalog, unito alla sua semplicità, ci permette di affrontare l'interpretazione di un buon numero di interrogazioni (semplici, ma molto frequenti) in linguaggio naturale.

7.4.1 Definire una forma logica per la domanda

Partendo dal grafo delle dipendenze e dalle annotazioni, costruiamo preliminarmente una rappresentazione della domanda denominata d'ora in avanti *programma logico intenzionale* o LIP.

Consideriamo come primo esempio l'interrogazione nq_1 sui ricercatori di intelligenza artificiale al DIMES:



La costruzione del programma logico inizia con la creazione di una regola logica, denominata *regola tematica*(topic rule), e da un insieme di regole logiche ausiliari. L'intenzione della domanda è definita, sulla base alle considerazioni linguistiche discusse nella precedente sezione, con l'elemento identificato come $Out(nq_1)$, nella fattispecie il nodo "who".[7] Si associa a questo nodo una variabile che diventa l'argomento di uno speciale predicato unario *return*. Tale predicato rappresenta la testa della regola tematica il cui corpo sarà composto da atomi associati al nodo *radice* nell'annotazione sintattica

[6] Utilizzare le diverse forme verbali per una migliore comprensione del contesto è tuttavia un tema interessante per futuri possibili miglioramenti.

[7] Per semplicità, ove non si genera confusione, useremo direttamente termini o sottostringhe in luogo di nodi del grafo delle dipendenze o sottoalberi.

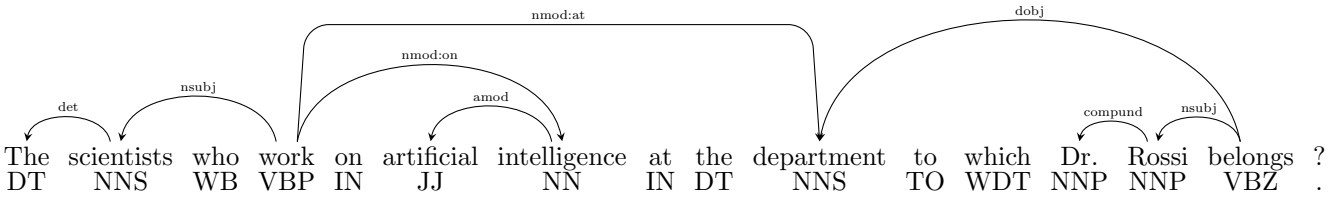
dell'interrogazione ed ai suoi dipendenti diretti. La regola tematica avrà quindi nel corpo tanti predicati unari quanti sono i dipendenti della radice. Tali predicati sono poi espansi in successive regole, chiamate *meaning auxiliary rule*(mar), nel cui corpo appariranno nuovi predicati la cui arità sarà incrementata di 1. In generale l'arità dei predicati che occorrono nelle regole del programma LIP dipendono dal livello del nodo a cui è associato il predicato che si sta definendo, mentre la lunghezza delle regole dipende dal numero di dipendenti di tale nodo.

Per facilitare la presentazione, il seguente algoritmo mostra la generazione della regola tematica, dalla quale, ricorsivamente, vengono lanciate le chiamate di generazione per le regole associate ai livelli successivi del grafo delle dipendenze. È importante notare che la generazione dei predicati e delle regole associate derivano, oltre che dall'annotazione sintattica (che distingue verbi, soggetti, o complementi), dall'eventuale tipo memorizzato nell'annotazione ANE. Ad esempio può essere necessario generare dei predicati speciali per conteggi finalizzati a confronti numerici, oppure gestire frasi collegate con una negazione.

Algoritmo 1: Costruttore regola tematica per interrogazioni

Data: Reduced dependency graph $G_{nq} = (N, D, pos, dep)$, Annotazioni ANE.
Result: LIP.
begin
  $LIP \longleftarrow$ *an empty logic program*
  $root \longleftarrow returnRootNode$
  $S \longleftarrow getDependents(root)$
  $V \longleftarrow$ *an empty set of variables*
  $X \longleftarrow selectOutVariable(ANE)$
  $main\_rule \longleftarrow append(\text{“}return(X) :\text{-”})$
  $V \longleftarrow V \cup X$
  for $d \in S$ **do**
    $p \longleftarrow$ *create* $Predicate(d, ANE, V)$
    append to $main\_rule(p)$
    $LIP \longleftarrow LIP \cup$ *create rules for* (p, ANE, V)
  Return LIP

Illustriamo il funzionamento dell'algoritmo ed introduciamo i passi seguenti con una variante leggermente arricchita del nostro running example.



Si tratta di una query la cui analisi sintattica individua una interrogazione di tipo "wh-" priva di confronti che coinvolgano conteggi o negazioni (o costrutti del tipo "only", "exactly", etc.). Stiamo quindi trattando una interrogazione con annotazioni *ane* molto semplici, più avanti saranno trattate le modalità di estensione della tecnica ad interrogazioni più complesse, lasciando comunque ad un lavoro futuro l'implementazione completa di tali estensioni.

Il nodo "scientists" viene annotato con il tipo variabile di output, rendendolo il tema della domanda. Sfruttando la relazione di dominanza si può scrivere la regola principale per la domanda come:

$$return(S) \coloneq scientist(S), work\_on_{intelligence_{artificial}}(S), \\ work\_at_{department_{Belongs_{Dr\_Rossi}}}(S).$$

tuttavia per migliorare la leggibilità può essere opportuno mantenere l'ordine presente nella domanda nei pedici dei predicati:

$$return(S) \coloneq scientist(S), work\_on_{artificial\_intelligence}(S), \\ work\_at_{department\_Dr\_Rossi\_Belongs}(S).$$

Esplicitare il ruolo del modificatore aggettivale

Uno dei principali tipi di predicato che viene generato è quello associato ai modificatori in presenza di aggettivi. In particolare la dipendenza *amod* identifica un'espressione aggettivale che modifica il significato del nome. Come già discusso in precedenza, esistono non poche difficoltà nel generalizzare il contributo semantico fornito da un aggettivo nella definizione del significato di una struttura aggettivale. Sulla base di quelle considerazioni l'approccio perseguito qui non tenterà di scendere nei dettagli di una ricostruzione semantica che conduca alla risoluzione dell'esatta modellazione del legame fra nome e aggettivo, ma cercherà di catturare il significato dell'espressione aggettivale attraverso la ricerca del concetto risultante all'interno della base di conoscenza o, in caso di insuccesso, estendendo tale ricerca alle reti semantiche offerte da risorse esterne.

La gestione di tali modificatori introduce quindi incertezza a livello del programma LIP, poiché non è facile determinare se siamo in presenza di un concetto associato a qualche predicato o a qualche entità della base di conoscenza KB, ovvero a qualche "dato" concreto da individuare nella base

di dati estensionale di KB. In questa fase, entrambe le ipotesi vengono tenute attive e vengono quindi generate due regole per gestire questi predicati.

$work\_on_{artificial\_intelligence}(S) \text{:-} work\_on(S, I), intelligence_{artificial}(I).$
$work\_on_{artificial\_intelligence}(S) \text{:-} work\_on(S, I), \#like(I, \text{“}artificial\ intelligence''}).$
$intelligence_{artificial}(I) \text{:-} intelligence(I, A), artificial(A).$
$intelligence_{artificial}(I) \text{:-} intelligence(I, A), \#like(A, \text{“}artificial''}).$

La regola del secondo tipo (ad esempio la regola 2 nel caso di $work\_on_{artificial\_intelligence}(S)$) usa un predicato speciale, chiamato $\#like$, che viene implementato internamente con una procedura di *match* non solo letterale ma anche semantica. Tale *match* sarà effettuato tra la stringa elencata negli argomenti e i dati testuali che saranno individuati nell'estensione del predicato della KB che sarà associato a $work\_on_{artificial\_intelligence}(S)$ sfruttando le tecniche di vicinanza descritte nei precedenti capitoli e le parole di contesto associate alle annotazioni del predicato di destinazione.

Nel prototipo in fase di implementazione, questo predicato speciale viene risolto in fase di *grounding* della KB *dlv*. Verrà aggiunto un fatto del tipo $work\_on_{artificial\_intelligence}(\text{“}2571'')$ se il ricercatore identificato da "2571 si occupa esplicitamente di "artificial intelligence" oppure di un argomento che siamo stati in grado di identificare come tale. Nella versione corrente il fatto viene aggiunto se si supera una certa soglia numerica prefissata di vicinanza, tuttavia vi sono altre possibilità basate sulla minimizzazione di weak-constraint associati a tali fatti, che saranno approfonditi nel lavoro ancora in corso.

Utilizziamo la stessa tecnica per ulteriori tipologie simili alla modificazione aggettivale, ad esempio quelle etichettate dal parser sintattico come *compound*. È il caso della gestione dei nomi composti, come nell'espressione "*game theory*", ma anche di "*Dr. Rossi*" nel nostro running example:

$work\_at_{department\_Dr\_Rossi\_Belongs}(S) \text{:-} work\_at(S, I), Dr\_Rossi\_Belongs(I).$
$Dr\_Rossi\_Belongs(I) \text{:-} belongs(I, D), Dr\_Rossi(D).$
$Dr\_Rossi\_Belongs(I) \text{:-} belongs(I, D), \#like(D, \text{“}Dr.Rossi'').$

Si noti che $Dr\_Rossi(D)$ potrebbe essere in teoria oggetto di una ulteriore espansione in cui si inserisca una nuova regola il cui corpo contenga un predicato unario del tipo $Dr(D)$. Tale regola dovrebbe verificare una eventuale proprietà Dr presente in KB, codificata sotto forma di qualche predicato (non necessariamente unario) che possa essere fatto corrispondere a *Dr.* Nell'implementazione attuale tuttavia queste espansioni sono usate solo per le modificazioni aggettivali e non per le etichette di tipo *compound*.

Definizione dei predicati di LIP attraverso i predicati della KB

Come descritto nelle sezioni precedenti l'analisi dell'interrogazione produce un insieme di regole logiche basate su predicati *fresh* che nel programma LIP occorrono come predicati estensionali, ma che vanno poi definiti attraverso i predicati della base di conoscenza. In questo modo la valutazione del programma LIP insieme a KB produrrà la risposta desiderata (attraverso il predicato $return(\cdot)$).

La definizione di tali predicati richiede un'attenta valutazione poiché da un lato deve rispecchiare il più possibile l'intenzione dell'espressione linguistica in esame, dall'altro deve fornire un riferimento per i nodi del grafo attraverso gli elementi della base di conoscenza. Selezionare il corretto riferimento all'interno di una base di conoscenza con predicati n-ari richiede una diversa metodologia rispetto alle tecniche utilizzate per dati strutturati sul semplice paradigma entità-proprietà-entità. Nel nostro caso, infatti, la struttura delle rete offre poche indicazioni per esplicitare i legami fra un predicato ed i suoi argomenti e le corrispondenze necessarie devono essere individuate a partire dal modello di annotazione PSA.

Sia $pq(\bar{X})$ un predicato che occorre nel programma LIP come predicato estensionale, ad esempio $work\_on(S, I)$. In base all'analisi sintattica e per come esso è stato generato abbiamo a disposizione le seguenti informazioni:

Tipo predicato: può essere un predicato associato ad una forma verbale, oppure individuare una proprietà, etc. Nel caso di $work\_on(S, I)$ si tratta appunto di un predicato associato ad un verbo che corrisponde, per quanto riguarda le relazioni ConceptNet, con una relazione del tipo $CapableOf(S, I)$.

Contesto: è un insieme di termini lessicali costruiti a partire dal nome del predicato e dei suoi dipendenti diretti (inclusi i loro modificatori o compound). Le parole associate al predicato in esempio includono *work on, scientist, artificial intelligence*, oltre che eventuali parole individuate nelle risorse lessicali a disposizione (ad esempio BabelNet).

Ruoli: sono tipi funzionali quali *actor, object*, o *location*.

Sulla base di tali informazioni, l'obiettivo è di individuare un predicato target $pt(\bar{Y})$ nella base di conoscenza che possa essere usato per definire pq con una regola del tipo

$$pq(\bar{X}) \coloneq pt(\bar{Y}).$$

I due predicati in generale non hanno la stessa arità e le variabili $\bar{X}$ devono occorrere in $\bar{Y}$. In presenza di negazione, occorre in realtà scrivere due regole, poiché una delle due deve definire il predicato con il quale realizzare la negazione (solo stratificata). In questa presentazione per semplicità trascuriamo questo caso.

Occorre quindi individuare sia il predicato sia le variabili di interesse, più precisamente le posizioni in cui collocare le variabili $\bar{X}$ nel predicato pt della base di conoscenza. Per realizzare questo *match* usiamo le informazioni, simili

a quelle sopra esposte, predisposte per i predicati della KB sulla base delle annotazioni PSA. Ricordiamo infatti, dalla Sezione 7.2.1, che per ogni predicato $pt(\bar{Y})$ abbiamo la tipologia del predicato con le relazioni ConceptNet, le parole associate con i Synset di BabelNet, i ruoli di ciascun argomento, etc. In particolare, per ciascun predicato $pt(\bar{Y})$ di KB, il suo contesto è costituito dalle parole presenti nella sua annotazione a cui è possibile aggiungere gli iperonimi dei synset coinvolti.[8]

Per valutare la similarità semantica fra pq e ciascun predicato "compatibile" pt, utilizziamo le tecniche descritte nella Sezione 3.5. In particolare costruiamo i vettori di parole C_{pq} e C_{pt} associati ai due predicati e valutiamo la seguente misura di similarità:

$$semSim(C_{pt}, C_{pq}) = \sqrt[|C_{pt}|]{\prod_{c \in C_{pt}} \frac{\sum_{v \in C_{pq}} cos(c, v) + 1}{|C_{pq}|}} \tag{7.1}$$

La misura di similarità definita in 7.1 permette di indurre un'ordine di idoneità sui predicati della base di conoscenza. Questo tipo di approccio ci permette altresì di calcolare, invece del solo candidato più premettente, una lista dei primi k candidati più semanticamente simili.

A questo punto abbiamo a disposizione un certo numero di scelte per ciascun predicato e, se il numero non è eccessivo ed il tempo lo permette, è possibile considerare diverse combinazioni. Possiamo cioè valutare una misura di similarità semantica globale, tra i contesti C_{mar} dei predicati in LIP ed i contesti di ciascuna combinazione P di predicati target (o di un campionamento di combinazioni):

$$\begin{aligned} &\max_{P} \ semSim(C_{mar}, C_P) \\ &\text{s.t.} \qquad\qquad P \subseteq KB \\ &\qquad \forall pq \in pred(LIP) \ \exists \ pt \in P, pt \in Cand_{pq}, \end{aligned} \tag{7.2}$$

dove P è un multinsieme di predicati target avente la stessa cardinalità di C_{mar} e $Cand_{pq}$ contiene l'elenco dei predicati target idonei, sia dal punto di vista della tipologia di predicato sia dal punto di vista della similarità (si classifica cioè nei primi k o tra quelli che superano una data soglia).

Generalizzare un concetto

Come precedentemente descritto, per facilitare la risoluzione dei concetti attraverso il modello PSA le annotazioni contengono anche informazioni su eventuali tassonomie dei concetti presenti. Sulla base di tali indicazioni possiamo

[8] Per i predicati intensionali di KB è stata anche considerata l'estensione del contesto con le parole associata ai predicati presenti nel corpo delle regole che definiscono il predicato in questione. Questa strada è risultata però inutile nel corrente approccio poiché quei termini occorrono già nelle annotazioni che accompagnano il predicato. Ovviamente considerare il corpo delle regole sarebbe invece indispensabile in caso di annotazioni completamente automatiche.

ad esempio sapere che, per quanto riguarda la nostra base di conoscenza (non è detto sia vero in contesti più ampi),

$$scientist(X) :\!- professor(X).$$
$$scientist(X) :\!- researcher(X).$$

Tuttavia, poiché i termini impiegati nelle domande sono rivolti comunemente ad ottenere informazioni sulle singole istanze del predicato, il lessico impiegato risiede spesso al di sotto del livello di astrazione utilizzato per descrivere le situazioni rappresentate dai predicati. In altre parole, il lessico delle interrogazioni è in parte orientato alle istanze e non solo ai concetti coinvolti. È quindi utile un processo di astrazione che ponga sullo stesso piano i concetti utilizzati in modo da semplificare la procedura di allineamento descritta nella precedente sezione.

Tale processo di astrazione può essere affettuato attraverso le relazioni di iperonimia presenti nelle tassonomie delle reti semantiche. Infatti risalendo lungo la relazione *IsA* di una rete semantica come BabelNet o ConceptNet è possibile ricavare l'iperonimo di ogni concetto, ed utilizzarlo come elemento sul quale misurare la similarità semantica.

Si consideri, ad esempio, il concetto *artificial intelligence* presente nella domanda precedente. Nel nostro esempio tale concetto non è presente nell'annotazione lessicale del predicato in cui compaiono gli oggetti della ricerca. Tuttavia risalendo lungo la tassonomia si può incontrare il concetto di *computer science*, che a sua volta condurrà a concetti più generali:

$$\textit{computer science } \textbf{\textit{isA}} \ \{\textit{tecnology, discipline, formal science}\}$$

Fra questi concetti introdotti, il concetto *discipline* ha fra i suoi sinonimi le forme *field* e *subject* che forniscono un più alto valore di similarità semantica con l'annotazione del predicato *research* della base di conoscenza.

Remark 7.4. È importante notare la differenza sostanziale con il predicato *#like* descritto in precedenza: quel predicato serviva a trovare la corrispondenza reale nei dati e quindi a determinare la presenza finale nel risultato della query di un ricercatore oppure un altro tra quelli presenti in KB. La generalizzazione appena descritta ha invece il compito di permetterci di individuare il match corretto tra il predicato *work_on* presente in LIP ed il predicato target più appropriato, ad esempio il predicato *research* che collega scienziati e linee di ricerca.

Un aspetto importante riguarda la scelta del corretto numero di generalizzazioni da eseguire. Risolvere questo problema non è semplice. Risalire troppo lungo le relazioni *IsA* può infatti condurre a spiacevoli *derive* semantiche del significato di un elemento in un determinato contesto, che può tradursi in un'errata concettualizzazione. Per questi motivi il processo di generalizzazione è qui circoscritto in modo euristico a due livelli di distanza dal concetto

di partenza presente nella domanda: per ogni predicato pq della query, si aggiungono alla sua annotazione (e quindi al suo contesto) i synset di BabelNet che siano genitore e nonno del termine in esame lungo la relazione IsA.

Compatibilità degli argomenti

Selezionato l'allineamento tra i predicati, occorre individuare la corretta collocazione delle variabili create mediante la procedura descritta nelle precedenti sezioni. L'allineamento infatti riferisce solo un nodo ad un certo predicato senza specificare quale ruolo, o argomento, assegnare ad ogni variabile.

Ricordiamo che l'analisi sintattica ha guidato le modalità di collocazione delle variabili nel predicato in esame, diciamo pq, del programma LIP associato all'interrogazione. Nel caso del predicato $work\_on(S, I)$ nella query di esempio, abbiamo una relazione del tipo $CapableOf(S, I)$ in cui S è l'agente dell'azione (annotazione sintattica: $nsubj$) ed I l'oggetto dell'azione, nella fattispecie il tema di ricerca.

Tali informazioni vengono confrontate con le indicazioni contenute nell'annotazione PSA del predicato target pt, per le quali abbiamo a disposizione una descrizione accurata (supervisionata da umani, per costruzione) dei ruoli degli argomenti. Oltre alle annotazioni sintattiche, abbiamo infatti l'annotazione Syn e l'esplicitazione dei ruoli funzionali descritti dal predicato binario $hasFunctType$. Nell'esempio, il predicato target selezionato è *research* e l'annotazione PSA ci vincola all'associazione dell'agente (scienziato) con il primo argomento e all'associazione dell'oggetto dell'azione (il tema) con il secondo argomento, sicché nel programma finale LP sarà aggiunta la regola

$$work\_on(S, I) \coloneq research(S, I, \_).$$

Purtroppo i vincoli funzionali sopra descritti possono lasciare ancora margini di incertezza, sicché in generale occorre fare ricorso a tutte le informazioni in nostro possesso, in particolare ai vettori di parole associati alle informazioni di contesto nell'annotazione PSA di pt ed ai vettori di parole associati al predicato pq, corrispondente ad un certo nodo del grafo delle dipendenze della query, ed ai predicati relativi ai nodi collegati ad esso. Nella fattispecie, consideriamo anche i vettori di parole per i predicati adiacenti di LIP, segnatamente, $scientist(S)$ e $intelligence_{artificial}(I)$.

È quindi possibile utilizzare la metodologia adottata per la selezione del predicato a livello degli argomenti, descritta nella precedente sezione, combinando le misure di similarità basate su vettori di parole con la valutazione dei vincoli funzionali sopra descritta, basata su sintassi e descrizione dei ruoli degli argomenti.

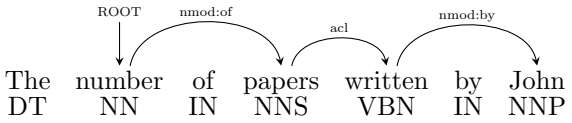
Osserviamo che i vincoli imposti sulle coppie dipendente-argomento possono essere estesi su diversi piani semantici oltre a quello funzionale e a quello ontologico. La scelta corrente è motivata dalla presenza di reti semantiche e di

strumenti che permettono un facile reperimento delle informazioni necessarie alla definizione dei vincoli.

Gestione dei predicati speciali

Come descritto nelle sezioni precedenti, durante la fase sintattica si assegnano alcuni particolari tipi alle annotazioni *ane* così da facilitare la costruzione dell'intenzione dell'interrogazione durante la fase interpretativa. Le espressioni "*how many*" e "*number of*", ad esempio, introducono rispettivamente i tipi *COUNT_G* e *COUNT_D*. Il controllo sull'eventuale presenza di questi e degli altri tipi riferiti a predicati speciali è condotta sulle regole *mar* subito dopo la fase di allineamento degli argomenti.

Si consideri la domanda descritta nella sezione precedente e riportata qui per comodità:



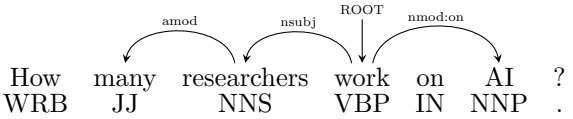
L'analisi sintattica assegna il tipo *COUNT_D* all'*ane* di "*number*", in modo da identificare la presenza di un predicato speciale. Più in dettaglio, al particolare tipo *COUNT_D* è associata la costruzione che prevede di introdurre all'interno della regola il predicato speciale $COUNT$, di utilizzare come suo argomento il dipendente legato attraverso la relazione *nmod:of*, ed infine sostituire nel predicato di testa il riferimento alla variabile associata al dipendente con la variabile generata dal predicato speciale:

$$number(X) \coloneq \#count\{P : papers(P)\} = X.$$
$$return(X) \coloneq number(X).$$

L'analisi del tipo non coinvolge in generale anche i predicati dipendenti. L'espressione "*how many*" produce lo stesso risultato di "*number of*" ma questa volta l'elemento da selezionare risulta essere il nodo governante. Questa piccola differenza impone una trattazione leggermente più complicata rispetto al caso precedente. Nel definire il conteggio sul predicato dipendente si sfruttano implicitamente le conseguenze del processo di costruzione dell'interpretazione: trovandosi ad un livello inferiore, la semantica del predicato è già definita al momento di diventare argomento del predicato speciale *COUNT*. Lo stesso non vale quando l'argomento è l'elemento governante. In questo caso infatti l'introduzione del predicato speciale non coinvolge la regola *mar* che contiene il tipo *COUNT_G* nel suo corpo, bensì una regola che può risiedere anche due livelli sopra. In altre parole, posto che esista una regola mar_{d_j} a livello l che contenga nel suo corpo un dipendente d_k il cui tipo è *COUNT_G*, allora la materializzazione dell'idea di conteggio si può concretizzare a livello

$l-2$ assegnando come argomento del predicato *COUNT* l'elemento nel corpo della regola d_i che governa d_j e discriminando sull'argomento assegnato a quest'ultimo. Infine la variabile assegnata alla funzione di conteggio sostituirà la variabile riferita a d_i nel predicato di testa.

Si consideri la seguente domanda:

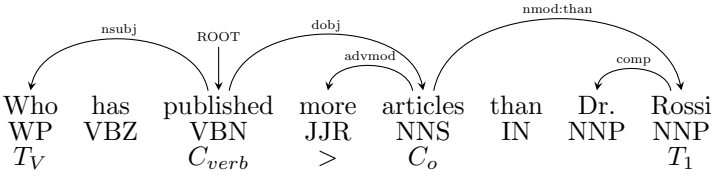


alla quale è possibile associare la seguente regola tematica:

$$return(X) \coloneq \#count\{R : researcher(R), work\_on\_AI(R)\} = X.$$

In questo caso la regola che riceve il predicato speciale è la regola tematica il cui predicato di testa *return* presenta la variabile X contenente il risultato del conteggio della funzione *COUNT*.

Nel caso di comparazione fra due entità della base di conoscenza la situazione è leggermente più complicata. Si consideri la domanda proposta nella precedente sezione e riportata qui per comodità. Essa pone in relazione il numero di articoli prodotti da *Dr. Rossi* col numero di articoli prodotto dagli altri autori della base di conoscenza.



Come descritto in precedenza il nodo *more*, sintatticamente annotato con JJR, è semanticamente valutato con il tipo $>$. Il suo nodo governante *articles* è selezionato come oggetto della comparazione poiché il suo tipo è C_o. L'espressione *published* ha come tipo C_{verb}, mentre i tipi per i termini da paragonare sono assegnati secondo la configurazione $\langle T_V, T_1 \rangle$.

Questa domanda è caratterizzata dalla presenza di un'espressione ellittica che attribuisce l'azione di pubblicare anche all'elemento *Rossi*. Per realizzare questa tipica attribuzione di azione al secondo termine del confronto si utilizza il predicato, selezionato per $mar_{C_{verb}}$, anche nella regola mar_{C_o} riferita all'oggetto del confronto. Si noti infine come quest'approccio implicitamente assegni lo stesso argomento ad entrambi i termini da confrontare. Ovvero per realizzare il confronto i termini coinvolti devono ricevere lo stesso argomento all'interno dello stesso predicato come loro riferimento, così come se si duplicasse la stessa azione della prima parte anche per "Dr. Rossi.

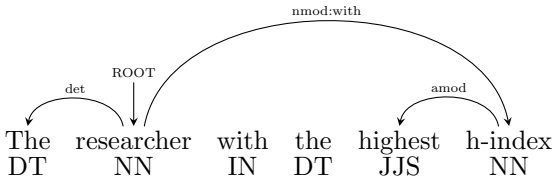
Selezionato il predicato all'interno della base di conoscenza, il confronto si concretizza materialmente all'interno della regola $mar_{C_{verb}}$. In questa regola il predicato selezionato diviene l'argomento della funzione di conteggio, e il risultato che produce tale funzione equivale al valore riferito al primo dei due

termini da comparare. Tuttavia nel diventare argomento della funzione speciale *COUNT* il predicato selezionato richiede di riferire l'argomento relativo al termine ad un ulteriore predicato della base di conoscenza. Assumendo che al soggetto sia attribuito il tipo T_V, il predicato da utilizzare è il predicato della base di conoscenza utilizzato nella regola riferita a T_1.

La seconda forma di conteggio è invece eseguita in mar_{C_o}. In questa regola si utilizza la stessa funzione di conteggio utilizzata in $mar_{C_{verb}}$, ma qui si può sfruttare la presenza del predicato riferimento al dipendente introdotto dalla preposizione *than*. Nell'esempio, il frammento di LIP con le regole che gestiscono il frammento col confronto è il seguente:

$$
\begin{aligned}
&return(W) \coloneq published\_art_m\_t\_rossi(W).\\
&published\_art_m\_t\_rossi(W) \coloneq published\_art_m(W, N_W),\\
&\qquad\qquad\qquad\qquad\quad published\_art_{rossi}(N_R), N_W > N_R.\\
&published\_art(W, X) \coloneq \quad \#count\{P : publish\_art(W, P)\} = X.\\
&published\_art_{rossi}(W, X) \coloneq \quad \#count\{P : publish\_art_{rossi}(P)\} = X.
\end{aligned}
$$

L'ultimo caso da trattare riguarda i costrutti con superlativo semplice. Si consideri la seguente domanda:



Qui l'elemento etichettato con JJS essendo semanticamente più vicino a massimo che a minimo riceve come tipo assegnato MAX. Questo tipo denota la funzione speciale che ritorna il massimo fra gli argomenti di un predicato. Come nel caso di *COUNT_G* l'argomento della funzione sarà il nodo governante il nodo JJS, ma qui la materializzazione accade nella regola immediatamente superiore: cioè in quella che contiene il nodo governante JJS fra i suoi dipendenti. Le regole generate sono allora:

$$
\begin{aligned}
&return(R) \coloneq researcher(R), highest\_h\_index(R).\\
&highest\_h\_index(R) \coloneq h\_index(R, H), max\_h\_index(H).\\
&max\_h\_index(H) \coloneq \#max\{Hi : h\_index(R, Hi)\} = H.
\end{aligned}
$$

Nelle regole riportate sopra è stata assunta l'esistenza di predicati con informazioni su ricercatori e *h-index*. Ad esempio, assumiamo che nel nostro caso vi sia un argomento del predicato *researcher* con le informazioni sui ricercatori che ne contiene l'h-index e che tale argomento sia individuato nell'annotazione del predicato come una proprietà del ricercatore (annotazione in Mod del tipo $hasProperty(researcher_{arg0}, researcher_{arg2})$). A causa della presenza del termine h-index nell'annotazione Syn di tale predicato, *researcher* sa-

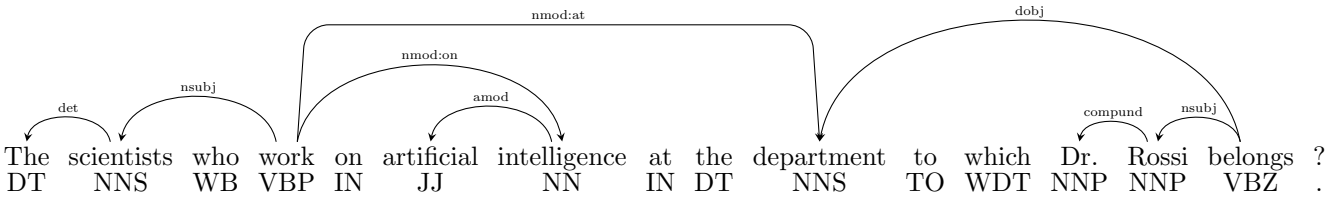
rebbe correttamente accoppiato al predicato $h\_index(R, Hi)$ che incontriamo nel programma LIP.

Ovviamente se nella KB non vi sono predicati con qualche informazione che possa essere accettabilmente associata ad *h_index* il sistema darà un messaggio di fallimento all'utente.

Costruzione del programma finale con l'output desiderato

Al termine delle analisi descritte nelle precedenti sezioni e della costruzione del programma intermedio LIP, occorre mettere insieme le regole di LIP e le regole di raccordo con i predicati della base di conoscenza per ottenere un unico programma DLP PQ in grado di rispondere all'interrogazione data sulla base di conoscenza KB. Oltre alla semplice unione delle regole descritte, occorre effettuare un ultimo passo elaborativo per tener conto che, tipicamente, le informazioni desiderate dagli utenti non corrispondono agli identificativi usati nei programmi per legare i predicati fra loro. Nel caso in cui l'interrogazione non elenchi in modo preciso le informazioni richieste (ad esempio chiedendo "nome, cognome, data di nascita"), il sistema deve individuare autonomamente un insieme di informazioni accettabile. Come già accennato all'inizio di questo capitolo, questa problematica viene qui risolta attraverso la predisposizione di alcuni predicati per l'output di default nella base di conoscenza, individuati appunto dal prefisso *default_*.

Per avere un'idea chiara del programma finale è utile riprendere una delle nostre query di esempio e mostrare cosa si ottiene combinando i vari pezzi che sono stati descritti nelle precedenti sezioni e prevedendo anche per l'output le informazioni desiderate.



Per le annotazioni della base di conoscenza della ricerca facciamo riferimento a quanto riportato in Appendice A (oltre a quanto è stato descritto in questo capitolo). Ricordiamo preliminarmente che il nodo "scientists" viene annotato con il tipo variabile di output, rendendolo il tema della domanda nella seguente regola tematica:

$$return(S) \text{ :- } scientist(S), work\_on_{artificial\_intelligence}(S), \\ work\_at_{department\_Dr\_Rossi\_Belongs}(S).$$

Poiché l'interrogazione non specifica le informazioni richieste per gli "scienziati" desiderati, occorre fare riferimento alle regole che definiscono i predicati

del tipo "$default\_$" in KB. Con le medesime tecniche usate per gli allineamenti degli altri predicati, nel nostro esempio è possibile associare per quanto riguarda l'output il predicato *member* con *scientist* e prenderne quindi le informazioni di default individuate dall'apposita regola:

$$default\_member(Name, Family) \coloneq member(Id, Name, Family, Email).$$

contenuta nella base di conoscenza.

A seguito di tale associazione, per gestire le informazioni di output per i dati individuati dal predicato *return*, aggiungeremo al programma PQ la regola:

$$default\_member(Name, Family) \coloneq return(S), \\ member(S, Name, Family, Email).$$

È tuttavia importante evidenziare che, per quest'ultimo step, l'associazione dei predicati non è davvero fondamentale, almeno nei casi in cui si possa assumere che gli insiemi di stringhe usati nella base di conoscenza per identificare le varie entità non abbiano sovrapposizioni indesiderate. In questa ipotesi, infatti, in caso di incertezza nell'associazione tra *member* e *scientist*, nulla ci vieta di aggiungere altre regole corrispondenti ad altri potenziali candidati, ad esempio le due regole

$$default\_visitor(Name, Family, University) \coloneq return(S), \\ visitor(S, Name, Family, Email, University).$$

$$default\_researchLine(Field, Description) \coloneq return(S), \\ researchLine(S, Type, Field, Description).$$

Se le istanze di $return(S)$ negli answer set di PQ saranno corrette, le estensioni di tali predicati nel modello conterranno le informazioni desiderate. In particolare otterremo nome e cognome nel caso di ricercatori, a cui si aggiunge anche l'università di provenienza nel caso di visitors. Per le linee di ricerca richiediamo invece in output il settore ed una descrizione. Tuttavia, per la query di esempio, la variabile S avrà solo mapping su identificatori di *scienziati*, per cui ci aspettiamo che gli answer set di PQ non contengano alcuna istanza di $default\_researchLine(Field, Description)$.

Ricordiamo ora le altre regole di PQ (sono elencate prima le regole di LIP e poi quelle aggiuntive di collegamento con i predicati di KB):

Tabella 7.3. Programma LIP per PQ

$work\_on_{artificial\_intelligence}(S) \text{:-} work\_on(S, I), intelligence_{artificial}(I).$
$work\_on_{artificial\_intelligence}(S) \text{:-} work\_on(S, I), \#like(I, \text{“}artificial\ intelligence'').$
$intelligence_{artificial}(I) \text{:-} intelligence(I, A), artificial(A).$
$intelligence_{artificial}(I) \text{:-} intelligence(I, A), \#like(A, \text{“}artificial'').$
$work\_at_{department\_Dr\_Rossi\_Belongs}(S) \text{:-} work\_at(S, I), Dr\_Rossi\_Belongs(I).$
$Dr\_Rossi\_Belongs(I) \text{:-} belongs(I, D), Dr\_Rossi(D).$
$Dr\_Rossi\_Belongs(I) \text{:-} belongs(I, D), \#like(D, \text{“}Dr.Rossi'').$

$scientist(S) \text{:-} research(S, \_, \_).$
$work\_on(S, I) \text{:-} research(S, I, \_).$
$work\_at(S, I) \text{:-} belongsTo(S, I, \_).$
$belongs(I, D) \text{:-} member(I, D, \_).$

8

Conclusioni e lavori in corso

L'interazione con basi di conoscenza attraverso interrogazioni in linguaggio naturale è stato il primo obiettivo di questo lavoro di tesi. Una parte consistente del lavoro ha riguardato lo studio della vasta letteratura nel campo dell'analisi del linguaggio naturale e delle sue varie sfaccettature, sia dal punto di vista sintattico sia dal punto di vista semantico.

Successivamente è stato approfondito il problema di rispondere ad una interrogazione in linguaggio naturale su una base di conoscenza, codificata in ASP e precedentemente annotata in modo semi-automatico. L'annotazione ci permette di inquadrare ogni predicato ed i suoi argomenti dal punto di vista dei loro ruoli, sia nell'ambito della base di conoscenza in esame, sia nell'ambito di risorse lessicali e semantiche di uso generale, quali BabelNet o ConceptNet.

Come caso di studio è stata considerata una base di conoscenza costruita a partire dalle informazioni sulle attività di ricerca condotte presso l'Università della Calabria e raccolte in una scheda, predisposta dal Ministero dell'Istruzione, dell'Università e della Ricerca, denominata SUA-RD. Tale base di conoscenza è stata poi semplificata per motivi di presentazione ed annotata in base alla metodologia descritta nel lavoro. Sono state anche condotte delle sperimentazioni per verificare la bontà dell'approccio proposto, almeno per interrogazioni semplici, di tipo "wh-" e prive di negazione.

Il lavoro presentato nella tesi non raggiunge ancora pienamente gli obiettivi, di medio-lungo termine, di ottenere un'interfaccia in linguaggio naturale per basi di conoscenza completamente implementata e testata su un'ampia gamma di interrogazioni. Gli esperimenti preliminari che sono stati condotti sono comunque promettenti e sono quindi di stimolo a proseguire sulla strada tracciata in questo lavoro.

Alla data di consegna della tesi, molto resta ancora da fare:

- gestire in modo adeguato tutti i predicati built-in a disposizione dei moderni sistemi di asnwer-set programming;
- costruire programmi LIP per interrogazioni con costrutti più complessi, che richiedano ad esempio forme di doppia negazione innestata;

- costruire programmi che permettano di gestire in modo automatico, attraverso una modellazione basata sui *weak-constraint*, diverse possibili interpretazioni dell'interrogazione, in caso di ambiguità nella sua formulazione in linguaggio naturale;
- completamento del prototipo e test completo sul caso di studio delle schede della ricerca dell'Università della Calabria da parte di diverse tipologie di utenti;
- versione multilingua in grado di gestire anche interrogazioni in lingua italiana, utilizzando le scarse risorse open-source disponibili per l'italiano, eventualmente combinate con tecniche di traduzione automatica in inglese, per beneficiare degli strumenti (ben più numerosi e completi) disponibili per l'inglese.

A

Esempio Annotazione PSA

Di seguito è riportato un piccolo esempio di annotazione per una base di conoscenza rivolta alla rappresentazione e alla gestione dell'informazione riguardante la ricerca accademica realizzata all'interno di una università. Si inizierà quindi presentando la struttura logica della basi di conoscenza e successivamente si descriveranno alcuni aspetti della fase di annotazione.

Le entità coinvolte nell'attività di ricerca sono esplicitate attraverso la definizione di determinati simboli di predicato che saranno utilizzati nella base di conoscenza DLV. Gli autori della ricerca sono divisi in due categorie: *member* e *visitor*. I primi rappresentano il personale interno all'università, mentre i secondi si riferiscono invece ai ricercatori in visita. La loro organizzazione all'interno dell'ateneo è realizzata attraverso una loro suddivisione in gruppi di ricerca, definiti da *researchGroup*. La definizione della ricerca è realizzata utilizzando il predicato *researchLine*, le cui estensioni saranno poi impiegate per caratterizzare gli articoli definiti con il predicato *paper*. Infine, il simbolo *externalResearchGroup* rappresenta informazioni su gruppi di ricerca esterni all'università che hanno collaborato con qualche gruppo interno. L'elenco completo dei predicati per le entità è riportato in tabella A.1.

Tabella A.1. Definizione delle entità coinvolte nell'attività di ricerca

Predicato
$member(Id, Name, Family, Email)$
$department(Id, Name, Address, Tel, Active)$
$researchGroup(Id, Name, Field, Description)$
$externalResearchGroup(Id, Name, Description, University)$
$visitor(Id, Name, Family, Email, University)$
$researchLine(Id, Type, Field, Description)$
$paper(Id, Title, First\_Author, Second\_Author, ResearchLine, Topic)$

Gli identificativi delle entità appena descritte sono gli argomenti delle relazioni d'interesse per il domino. Fra queste con i predicati *belongsTo* e *workFor* si modellano rispettivamente le relazioni di appartenenza e di affiliazione. La prima lega un membro dell'ateneo ad un dipartimento e ne definisce il ruolo fra $\{phdStudent, researcher, associate, professor\}$, mentre il secondo stabilisce l'affiliazione di un membro ad un gruppo di ricerca. La partecipazione di un visitatore esterno a qualche attività di ricerca è invece rappresentata dal simbolo di predicato *visitingIn*. Le ulteriori relazioni presenti nella base di conoscenza sono: *collaboration* per descrivere la collaborazione fra un gruppo di ricerca interno ed uno appartenente ad un'altra università, e *visitingOut* che rappresenta l'attività di ricerca condotta da un membro interno in una sede esterna. Infine l'attività di pubblicazione è rappresentata dalla relazione *publish*. La tabella A.2 mostra i dettagli per ognuna delle relazioni appena discusse.

Tabella A.2. Definizione delle relazioni fra le entità dell'attività di ricerca

Predicato
$workFor(Researcher, ResearchGroup)$
$belongsTo(Academician, Department, Degree)$
$publish(Researcher, Paper, Journal, Year)$
$visitingIn(visitor, ResearchGroup, Topic, Start, End)$
$visitingOut(Researcher, Host, Topic, Start, End)$
$collaboration(ResearchGroup, ExternalResearchGroup, Topic, Start, End)$

L'ultima parte della definizione della base di conoscenza riguarda la definizione dei predicati intensionali per esplicitare aspetti d'interesse. Per semplicità, in questo esempio, si introduce un solo predicato intensionale *research* per esplicitare il legame fra ogni ricercatore (indifferentemente se sia un membro interno o un visitatore) ed un particolare gruppo di ricerca, in riferimento ad un determinato argomento di ricerca. Definire questa relazione è immediato:

$$\begin{aligned} research(Researcher,\ & ResearchLine, ResearchGroup) \coloneq \\ & publish(Researcher, Paper, \_, \_), \\ & paper(Paper, \_, \_, \_, ResearchLine, \_), \\ & workFor(Researcher, ResearchGroup). \end{aligned}$$

Prima di discutere l'annotazione PSA si definiscono rapidamente una serie di predicati ausiliari utilizzati in fase di generazione di risposta. Al livello sintattico tutti questi predicati sono caratterizzati dal suffisso "*default_*" ed hanno lo scopo di selezionare l'informazione, più frequentemente richiesta, fra quelle presenti nei predicati riferiti alle entità. La tabella A.3 mostra l'insieme dei predicati ausiliari definiti per le entità della base di conoscenza.

Tabella A.3. Definizione delle entità coinvolte nell'attività di ricerca

Predicato
$default\_member(Name, Family) \text{ :- } member(Id, Name, Family, Email)$
$default\_department(Name) \text{ :- } department(Id, Name, Address, Tel, Active)$
$default\_researchGroup(Name, Field) \text{ :- } researchGroup(Id, Name, Field, Description)$
$default\_externalResearchGroup(Name, University) \text{ :- } externalResearchGroup(Id, Name, Description, University)$
$default\_visitor(Name, Family, University) \text{ :- } visitor(Id, Name, Family, Email, University)$
$default\_researchLine(Field, Description) \text{ :- } researchLine(Id, Type, Field, Description)$
$default\_paper(Title, First\_Author, Second\_Author) \text{ :- } paper(Id, Title, First\_Author, Second\_Author, ResearchLine, Topic)$

L'annotazione PSA è principalmente rivolta all'arricchimento semantico dei predicati che definiscono le relazioni fra le entità della base di conoscenza. Tali predicati possono infatti esprimere diverse dinamiche semantiche che caratterizzano la situazione descritta dal predicato e che deve essere resa esplicita per migliorare le prestazioni del sistema. Per semplicità di presentazione nel seguito la discussione sarà ristretta esclusivamente a questa particolare tipologia di predicati, tralasciando le annotazioni sui predicati che caratterizzano le proprietà delle entità.

Il primo predicato annotato secondo la metodologia proposta in PSA è *workFor(Researcher, ResearchGroup)*. Questo predicato binario definisce l'appartenenza di un ricercatore ad un determinato gruppo di ricerca. La tripla $\langle Lex, Syn, Mod \rangle$ relativa alla sua annotazione PSA è mostrata in A.4.

Tabella A.4. Annotazione PSA per il predicato *workFor(Researcher, ResearchGroup)*

Lex	$\langle''work'', v_{work},'' bn : 00095812v''\rangle$, $\langle''researcher'', v_{researcher},'' bn : 00047356n''\rangle$, $\langle''group', v_{group},'' bn : 00041942n''\rangle$
Syn	*"A scientist is member of research group" :* $\{(''scientist'', 1), (''is\ member\ of'', 0), (''research\ group'', 2)\}$; *"A researcher works in a group" :* $\{(''researcher'', 1), (''works'', 0), (''group'', 2)\}$
Mod	$isA(workFor_{arg1}, bn00047356n)$. $isA(workFor_{arg2}, bn00041942n)$. $partOf(workFor_{arg1}, workFor_{arg2})$.

Tabella A.5. Annotazione PSA per il predicato *belongsTo(Academician, Department, Degree)*

Lex	$\langle''work'', v_{work},'' bn : 00095812v''\rangle$, $\langle''academician'', v_{academician},'' bn : 00000551n''\rangle$, $\langle''department'', v_{dept},'' bn : 00026308n''\rangle$, $\langle''academic\ degree', v_{ad},'' bn : 00000554n''\rangle$
Syn	*"A professor works in a department" :* $\{(''professor'', 1), (''works\ in'', 0), (''department'', 2)\}$; *"A academician has a degree inside his Dept" :* $\{(''academician'', 1), (''Dept'', 2), (''degree'', 3)\}$
Mod	$isA(belongsTo_{arg1}, bn00000551n)$. $isA(belongsTo_{arg2}, bn00026308n)$. $isA(belongsTo_{arg3}, bn00000554n)$. $hasA(belongsTo_{arg1}, belongsTo_{arg3})$. $RelatedTo(belongsTo_{arg1}, belongsTo_{arg2})$

Tabella A.6. Annotazione PSA per il predicato *publish(Researcher, Paper, Journal, Year)*

Lex	$\langle''publish'', v_{publish},'' bn : 00092123v''\rangle$, $\langle''researcher'', v_{researcher},'' bn : 00047356n''\rangle$, $\langle''paper', v_{paper},'' bn : 00060466n''\rangle\}$ $\langle''journal', v_{journal},'' bn : 00048453n''\rangle\}$ $\langle''year', v_{year},'' bn : 00078738n''\rangle$
Syn	*"Researchers publish their work on some journal in 2016." :* $\{(''researchers'', 1), (''publish'', 0), (''work'', 2), (''journal'', 3), (''2016'', 4)\}$; *"An article is presented by the authors in a conference" :* $\{(''author'', 1), (''presented'', 0), (''article'', 2), (''conference'', 3)\}$
Mod	$isA(publish_{arg0}, bn00092123v)$. $isA(publish_{arg1}, bn00047356n)$. $isA(publish_{arg2}, bn00060466n)$. $isA(publish_{arg3}, bn00048453n)$. $isA(publish_{arg4}, bn00078738n)$. $capableOf(publish_{arg1}, publish_{arg0})$. $createdBy(publish_{arg2}, publish_{arg1})$. $atLocation(publish_{arg2}, publish_{arg3})$. $hasProperty(publish_{arg0}, publish_{arg4})$. $RelatedTo(publish_{arg0}, publish_{arg2})$.

Tabella A.7. Annotazione PSA per il predicato *research(Researcher, ResearchLine, ResearchGroup)*

Lex	$\langle''research'', v_{research},'' bn : 00095823v''\rangle$, $\langle''researcher'', v_{researcher},'' bn : 00047356n''\rangle$, $\langle''topic', v_{topic},'' bn : 00074900n''\rangle$ $\langle''department'', v_{dept},'' bn : 00026308n''\rangle$,
Syn	*"A scientist of reseach group works on a topic." :* $\{(''scientist'', 1), (''works\ on'', 0), (''topic'', 2), (''reseach\ group'', 3)\}$;
Mod	$isA(research_{arg0}, bn00095823v)$. $isA(research_{arg1}, bn00047356n)$. $isA(research_{arg2}, bn00074900n)$. $isA(research_{arg3}, bn00026308n)$. $capableOf(research_{arg1}, research_{arg0})$. $hasProperty(research_{arg0}, research_{arg2})$. $relatedTo(research_{arg0}, research_{arg2})$. $atLocation(research_{arg0}, research_{arg3})$.

A

Relazioni semantiche

In entrambe le risorse, WordNet e BabelNet, le relazioni fra un synset e altri synset sono descritte da puntatori. Quest'ultimi possono essere descritti come lessicali o semantici. I primi rappresentano relazioni fra forme di parole, e mettono in relazione esclusivamente specifiche parole all'interno di entrambi i synset. In questa categoria rientrano le relazioni di Antonym, Hypernymy/Hyponymy,Pertainym, Participle, Also See, Derivationally Related. Mentre le restanti categorie sono generalmente classificate come relazioni semantiche e si riferiscono a tutte le parole contenute nei synset.

A.1 WordNet

In WordNet le relazioni fra un synset sorgente ed uno target sono realizzate specificando le parole nel caso di relazioni lessicali o i synset stessi seguiti da un simbolo che rappresenta la tipologia di relazione. Di seguito sono riportati le tipologie di relazioni permesse per le diverse categorie sintattiche.

Tabella A.1. WordNet : puntatori per i nomi

Simbolo	Tipo	Descrizione
!	Antonym	Una coppia di parole fra le quali esiste una relazione contraddittorie nella loro applicazione
@	Hypernym	Il termine generico utilizzato per definire una classe di specifiche istanze. Y è l'iperonimo di X se X è una specie di Y.
@i	Instance Hypernym	
~	Hyponym	Un termine specifico per identificare un membro di una classe.
~i	Instance Hyponym	
#m	Member holonym	
#s	Substance holonym	
#p	Part holonym	
%m	Member meronym	Il nome di un costituente membro di qualcosa. X è meronimo di Y se X è parte di Y.
%s	Substance meronym	Il nome del costituente in termini di sostanza.
%p	Part meronym	Il nome della parte costituente qualcosa.
=	Attribute	Un nome per il quale gli aggettivi esprimono i valori.
+	Derivationally related form	Termini in differenti sintattiche categorie che hanno la stessa forma canonica e sono semanticamente collegati.
;c	Domain of synset - TOPIC	Classificazione topica nella quale il synset utilizza un puntatore *TOPIC*.
-c	Member of this domain - TOPIC	
;r	Domain of synset - REGION	Classificazione topica nella quale il synset utilizza un puntatore *REGION*.
-r	Member of this domain - REGION	
;u	Domain of synset - USAGE	Classificazione topica nella quale il synset utilizza un puntatore *USAGE*.
-u	Member of this domain - USAGE	

[a] Table foot note (with superscript)

Tabella A.2. WordNet : puntatori per i verbi, aggettivi, e avverbi.

Simbolo	Tipo	Simbolo	Tipo	Simbolo	Tipo
!	Antonym	!	Antonym	!	Antonym
@	Hypernym	&	Similar to		Derived from adjective
~	Hyponym	¡	Participle of verb	;c	Domain of synset - TOPIC
	Entailment		Pertainym (pertains to noun)	;r	Domain of synset - REGION
¿	Cause	=	Attribute	;u	Domain of synset - USAGE
	Also see		Also see		
$	Verb Group	;c	Domain of synset - TOPIC		
+	Derivationally related form	;r	Domain of synset - REGION		
;c	Domain of synset - TOPIC	;u	Domain of synset - USAGE		
;r	Domain of synset - REGION				
;u	Domain of synset - USAGE				

[a] Table foot note (with superscript)

Tabella A.3. Elenco delle relazioni utilizzate nella rete BabelNet

Tipo	Descrizione
ALSO_SEE	Also See da WordNet
ANTONYM	Antonym da WordNet
ANY_HOLONYM	Holonyms da all resources
ANY_HYPERNYM	Hypernyms da all resources
ANY_HYPONYM	Hyponyms da all resources
ANY_MERONYM	Meronyms da all resources
ATTRIBUTE	Attribute da WordNet
CAUSE	Cause da WordNet
DERIVATIONALLY_RELATED	Derivationally related form da WordNet
ENTAILMENT	Entailment da WordNet
GLOSS_DISAMBIGUATED	Gloss related form (disambiguated) da WordNet
GLOSS_MONOSEMOUS	Gloss related form (monosemous) da WordNet
HOLONYM_MEMBER	Member holonym da WordNet
HOLONYM_PART	Part holonym da WordNet
HOLONYM_SUBSTANCE	Substance holonym da WordNet
HYPERNYM	Hypernym da WordNet
HYPERNYM_INSTANCE	Instance hypernym da WordNet
HYPONYM	Hyponym da WordNet
HYPONYM_INSTANCE	Instance hyponym da WordNet
MERONYM_MEMBER	Member meronym da WordNet
MERONYM_PART	Part meronym da WordNet
MERONYM_SUBSTANCE	Substance meronym da WordNet
PARTICIPLE	Participle da WordNet
PERTAINYM	Pertainym da WordNet
REGION	Domain of synset - REGION da WordNet
REGION_MEMBER	Member of this domain da WordNet
SEMANTICALLY_RELATED	Wikipedia relations
SIMILAR_TO	Similar To da WordNet
TOPIC	Domain of synset - TOPIC da WordNet
TOPIC_MEMBER	Member of this domain - TOPIC da WordNet
USAGE	Domain of synset - USAGE da WordNet
USAGE_MEMBER	Member of this domain - USAGE da WordNet
VERB_GROUP	Verb Group da WordNet
WIBI_HYPERNYM	Hypernym da Wikipedia Bitaxonomy
WIBI_HYPONYM	Hyponym da Wikipedia Bitaxonomy
WIKIDATA_HYPERNYM	Hypernym da Wikidata
WIKIDATA_HYPONYM	Hyponym da Wikidata
WIKIDATA_MERONYM	Meronym da Wikidata

[a] Table foot note (with superscript)

Tabella A.4. Elenco delle relazioni utilizzate nella rete ConceptNet

URI	Descrizione	Esempio
/r/RelatedTo	La più generale fra le relazioni . Esiste qualche positiva relazione fra A e B, ma ConceptNet non può stabilire di che che natura sia basandosi sui dati. tipologia simmetrica.	learn ↔ erudition
/r/ExternalURL	Puntatore ad un URL esterno a ConceptNet.	knowledge→ http://dbpedia.org/page/Knowledge
/r/FormOf	A è una qualche forma inflessa di B; B non è la radice di A.	slept → sleep
/r/IsA	A è sottotipo o una specifica istanza di B; qualsiasi A è una B. Ciò può includere specifiche istanze benché tale distinzione può essere poco evidente nel linguaggio. Corrisponde alla relazione *hyponym* di WordNet.	car → vehicle; Chicago → city
/r/PartOf	A è parte di B. Rappresenta la relazione di PartMeronym di WordNet.	gearshift → car
/r/HasA	B appartiene ad A, o come sua parte costituente o per qualche socialmente accettato espressione di possesso. HasA è spesso l'inversa di PartOf.	bird → wing; pen → ink
/r/UsedFor	A è usato per B; lo scopo di A è B.	bridge → cross water
/r/CapableOf	Qualcosa che A può tipicamente fare è B.	knife → cut butter → refrigerator;
/r/AtLocation	A è una tipica locazione di B, o A è la locazione ereditata da B.	Boston → Massachusetts
/r/Causes	A e B sono eventi, e tipicamente A causa B.	exercise → sweat
/r/HasSubevent	A e B sono eventi, e B accade come sotto-evento di A.	eating → chewing
/r/HasFirstSubevent	A è un evento che inizia con il sotto-evento B.	sleep → close eyes
/r/HasLastSubevent	A è un evento che si conclude con il sotto-evento B.	cook → clean up kitchen
/r/HasPrerequisite	Affinché A abbia luogo, B deve accadere.	dream → sleep
/r/HasProperty	A ha B come sua proprietà; A può essere descritto come B.	ice → cold
/r/MotivatedByGoal	Qualcuno compie A per ottenere il risultato B; A è qualche passo verso il raggiungimento di B.	compete → win
/r/ObstructedBy	A è un obbiettivo che può essere precluso da B; B ostacola in qualche modo A.	sleep → noise
/r/Desires	A è un'entità cosciente che tipicamente vuole B. Molte asserzioni di questo tipo usano person come A.	person → love
/r/CreatedBy	B è un processo o un agente che crea A.	cake → bake
/r/Synonym	A e B hanno un significato molto simile. Simmetrica.	sunlight ↔ sunshine
/r/Antonym	A e B sono opposti in qualche modo rilevante, come l'inizio è opposto alla fine di una scala, o cose simili che hanno caratteristiche fondamentali differenti. Simmetrica.	black ↔ white; hot ↔ cold
/r/DerivedFrom	A è una parola o frase che appare dentro B e contribuisce il significato di B.	pocketbook → book
/r/SymbolOf	A rappresenta simbolicamente B.	red → fervor
/r/DefinedAs	A e B si sovrappongono nel significato, e B è più eloquente di A.	peace → absence of war
/r/Entails	Se A sta accadendo, B accade pure.	run → move
/r/MannerOf	A è uno specifico modo di fare B. Simile a IsA, ma riferito soprattutto ai verbi.	auction → sale
/r/LocatedNear	A e B sono tipicamente trovati vicini fra di loro. Simmetrica.	chair ↔ table

A

Dipendenze universali

Tabella A.1. Elenco delle dipendenze universali

Relazione	Descrizione
acl	clausal modifier of noun (adjectival clause)
advcl	adverbial clause modifier
advmod	adverbial modifier
amod	adjectival modifier
appos	appositional modifier
aux	auxiliary
auxpass	passive auxiliary
case	case marking
cc	coordinating conjunction
ccomp	clausal complement
compound	compound
conj	conjunct
cop	copula
csubj	clausal subject
csubjpass	clausal passive subject
dep	unspecified dependency
det	determiner
discourse	discourse element
dislocated	dislocated elements
dobj	direct object
expl	expletive
foreign	foreign words
goeswith	goes with
iobj	indirect object
list	list
mark	marker
mwe	multi-word expression
name	name
neg	negation modifier
nmod	nominal modifier
nsubj	nominal subject
nsubjpass	passive nominal subject
nummod	numeric modifier
parataxis	parataxis
punct	punctuation
remnant	remnant in ellipsis
reparandum	overridden disfluency
root	root
vocative	vocative
xcomp	open clausal complement

Bibliografia

1. Chitta Baral. *Knowledge representation, reasoning and declarative problem solving*. Cambridge university press, 2003.
2. Chitta Baral and Michael Gelfond. Logic programming and knowledge representation. *The Journal of Logic Programming*, 19:73–148, 1994.
3. Holger Bast, Fabian Suchanek, and Ingmar Weber. Semantic full-text search with ester: scalable, easy, fast. In *2008 IEEE International Conference on Data Mining Workshops*, pages 959–962. IEEE, 2008.
4. Rachel Ben-Eliyahu and Rina Dechter. Propositional semantics for disjunctive logic programs. *Annals of Mathematics and Artificial intelligence*, 12(1-2):53–87, 1994.
5. Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *journal of machine learning research*, 3(Feb):1137–1155, 2003.
6. Yoshua Bengio, Yann LeCun, et al. Scaling learning algorithms towards ai.
7. Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, volume 2, page 6, 2013.
8. Patrick Blackburn and Johan Bos. Representation and inference for natural language. *A first course in computational semantics. CSLI*, 2005.
9. David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
10. Stefano Borgo and Claudio Masolo. Ontological foundations of dolce. In *Theory and applications of ontology: Computer applications*, pages 279–295. Springer, 2010.
11. Johan Bos. Expressive power of abstract meaning representations. *Computational Linguistics*, 2016.
12. Stefan Brass and Jürgen Dix. Disjunctive semantics based upon partial and bottom-up evaluation. In *ICLP*, pages 199–213, 1995.
13. Michel Bréal. *Essai de sémantique:(science des significations)*. Hachette, 1904.
14. Joan Bresnan, Ronald M Kaplan, Stanley Peters, and Annie Zaenen. Cross-serial dependencies in dutch. In *The formal complexity of natural language*, pages 286–319. Springer, 1982.
15. Francesco Buccafurri, Nicola Leone, and Pasquale Rullo. Enhancing disjunctive datalog by constraints. *IEEE Transactions on Knowledge and Data Engineering*, 12(5):845–860, 2000.

16. Sabine Buchholz and Erwin Marsi. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 149–164. Association for Computational Linguistics, 2006.
17. Marco Cadoli, Thomas Eiter, and Georg Gottlob. Default logic as a query language. *IEEE Transactions on Knowledge and Data Engineering*, 9(3):448–463, 1997.
18. Francesco Calimeri, Martin Gebser, Marco Maratea, and Francesco Ricca. Design and results of the fifth answer set programming competition. *Artificial Intelligence*, 231:151–181, 2016.
19. José Camacho-Collados, Ignacio Iacobacci, Roberto Navigli, and Mohammad Taher Pilehvar. Semantic representations of word senses and concepts. *arXiv preprint arXiv:1608.00841*, 2016.
20. Rudolf Carnap. Modalities and quantification. *The Journal of Symbolic Logic*, 11(02):33–64, 1946.
21. Rudolf Carnap. On the application of inductive logic. *Philosophy and phenomenological research*, 8(1):133–148, 1947.
22. Danqi Chen and Christopher D Manning. A fast and accurate dependency parser using neural networks. In *EMNLP*, pages 740–750, 2014.
23. Noam Chomsky. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124, 1956.
24. Noam Chomsky. On certain formal properties of grammars. *Information and control*, 2(2):137–167, 1959.
25. Noam Chomsky. *Lectures on government and binding: The Pisa lectures*. Number 9. Walter de Gruyter, 1993.
26. Noam Chomsky. *Syntactic structures*. Walter de Gruyter, 2002.
27. Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.
28. James R Curran, Stephen Clark, and Johan Bos. Linguistically motivated large-scale nlp with c&c and boxer. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 33–36. Association for Computational Linguistics, 2007.
29. Mary Dalrymple. *Lexical-Functional Grammar*. Wiley Online Library, 2001.
30. Evgeny Dantsin, Thomas Eiter, Georg Gottlob, and Andrei Voronkov. Complexity and expressive power of logic programming. *ACM Computing Surveys (CSUR)*, 33(3):374–425, 2001.
31. Donald Davidson. Reply to quine. *Actions and events: Perspectives on the philosophy of Donald Davidson. Oxford: Blackwell*, 1986.
32. Donald Davidson. *Essays on actions and events: Philosophical essays*, volume 1. Oxford University Press on Demand, 2001.
33. Ralph Debusmann, Denys Duchier, and Geert-Jan M Kruijff. Extensible dependency grammar: A new methodology. In *Proceedings of the COLING 2004 Workshop on Recent Advances in Dependency Grammar*, pages 70–76, 2004.
34. Jürgen Dix, Georg Gottlob, and Wiktor Marek. Reducing disjunctive to non-disjunctive semantics by shift-operations. *Fundamenta Informaticae*, 28(1, 2):87–100, 1996.
35. David Dowty. Thematic proto-roles and argument selection. *Language*, pages 547–619, 1991.

36. Jason M Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 340–345. Association for Computational Linguistics, 1996.
37. Thomas Eiter, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Declarative problem-solving using the dlv system. In *Logic-based artificial intelligence*, pages 79–103. Springer, 2000.
38. Thomas Eiter, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Computing preferred answer sets by meta-interpretation in answer set programming. *Theory and Practice of Logic Programming*, 3(4+ 5):463–498, 2003.
39. Thomas Eiter and Georg Gottlob. On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence*, 15(3-4):289–323, 1995.
40. Thomas Eiter, Georg Gottlob, and Heikki Mannila. Disjunctive datalog. *ACM Transactions on Database Systems (TODS)*, 22(3):364–418, 1997.
41. Thomas Eiter, Nicola Leone, and Domenico Sacca. On the partial semantics for disjunctive deductive databases. *Annals of Mathematics and Artificial Intelligence*, 19(1-2):59–96, 1997.
42. Christiane Fellbaum, Derek Gross, and Katherine Miller. Adjectives in wordnet. 1993.
43. David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.
44. Charles J Fillmore. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences*, 280(1):20–32, 1976.
45. John Firth. A synopsis of linguistic theory, 1930-1955. *Studies in Linguistic Analysis*, pages 1–32, 1957.
46. Tiziano Flati and Roberto Navigli. The wikipedia bitaxonomy explorer. In *Proceedings of the 2014 International Conference on Posters & Demonstrations Track-Volume 1272*, pages 105–108. CEUR-WS. org, 2014.
47. Tiziano Flati, Daniele Vannella, Tommaso Pasini, and Roberto Navigli. Multiwibi: The multilingual wikipedia bitaxonomy project. *Artificial Intelligence*, 241:66–102, 2016.
48. Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.
49. Gottlob Frege. Sense and reference. *The philosophical review*, 57(3):209–230, 1948.
50. Robert Gaizauskas and Kevin Humphreys. A combined ir/nlp approach to question answering against large text collections. In *Content-Based Multimedia Information Access-Volume 2*, pages 1288–1304. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, 2000.
51. Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. Sweetening wordnet with dolce. *AI magazine*, 24(3):13, 2003.
52. Gerald Gazdar. *Generalized phrase structure grammar*. Harvard University Press, 1985.
53. Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New generation computing*, 9(3-4):365–385, 1991.

54. Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.
55. Bert F Green Jr, Alice K Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: an automatic question-answerer. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*, pages 219–224. ACM, 1961.
56. Thomas L Griffiths and Mark Steyvers. A probabilistic approach to semantic representation. In *Proceedings of the 24th annual conference of the cognitive science society*, pages 381–386. Citeseer, 2002.
57. Thomas R Gruber et al. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.
58. Nicola Guarino. Formal ontology and information systems. In *Proceedings of FOIS*, volume 98, pages 81–97, 1998.
59. Sanda M Harabagiu, Dan I Moldovan, Christine Clark, Mitchell Bowden, Andrew Hickl, and Patrick Wang. Employing two question answering systems in trec 2005. In *TREC*, 2005.
60. Sanda M Harabagiu, Dan I Moldovan, Marius Paşca, Rada Mihalcea, Mihai Surdeanu, Răzvan Bunescu, Corina R Gîrju, Vasile Rus, and Paul Morărescu. Falcon: Boosting knowledge for answer engines. 2000.
61. Z. S. Harris. *Mathematical Structures of Language*. Wiley, New York, NY, USA, 1968.
62. Peter Hellwig. Dependency unification grammar. In *Proceedings of the 11th coference on Computational linguistics*, pages 195–198. Association for Computational Linguistics, 1986.
63. Michael Hess. Reference and quantification in discourse. *University of Zurich Habilitation Thesis*, 1989.
64. Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.
65. John E Hopcroft. *Introduction to Automata Theory, Languages and Computation: For VTU, 3/e*. Pearson Education India, 1979.
66. Richard Hudson. *Language networks: the new Word Grammar*. Oxford University Press, 2007.
67. Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. Sensembed: learning sense embeddings for word and relational similarity. In *Proceedings of ACL*, pages 95–105, 2015.
68. Timo Jarvinen and Pasi Tapanainen. Towards an implementable dependency grammar. In *Proceedings of the workshop on processing of dependency-based grammars*, volume 10. Citeseer, 1998.
69. RichardSocher JeffreyPennington and ChristopherD Manning. Glove: Global vectors for word representation.
70. Aravind K Joshi. An introduction to tree adjoining grammars. *Mathematics of language*, 1:87–115, 1987.
71. Aravind K Joshi, Leon S Levy, and Masako Takahashi. Tree adjunct grammars. *Journal of computer and system sciences*, 10(1):136–163, 1975.
72. Hans Kamp. Evénements, représentations discursives et référence temporelle. *Langages*, (64):39–64, 1981.

73. Hans Kamp and Uwe Reyle. *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*, volume 42. Springer Science & Business Media, 2013.
74. Stig Kanger. A note on quantification and modalities. 1957.
75. Stig Kanger. Provability in logic. 1957.
76. Boris Katz, Gary C Borchardt, and Sue Felshin. Natural language annotations for question answering. In *FLAIRS Conference*, pages 303–306, 2006.
77. Jerrold J Katz and Jerry A Fodor. The structure of a semantic theory. *language*, 39(2):170–210, 1963.
78. Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. A large-scale classification of english verbs. *Language Resources and Evaluation*, 42(1):21–40, 2008.
79. Saul A Kripke. A completeness theorem in modal logic. *The journal of symbolic logic*, 24(01):1–14, 1959.
80. Saul A Kripke. Semantical analysis of modal logic i normal modal propositional calculi. *Mathematical Logic Quarterly*, 9(5-6):67–96, 1963.
81. Thomas K Landauer and Susan T Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211, 1997.
82. Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 704–710. Association for Computational Linguistics, 1998.
83. Alex Lascarides and Nicholas Asher. Segmented discourse representation theory: Dynamic semantics with discourse structure. In *Computing meaning*, pages 87–124. Springer, 2008.
84. Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Thomas Eiter, Georg Gottlob, Simona Perri, and Francesco Scarcello. The dlv system for knowledge representation and reasoning. *ACM Transactions on Computational Logic (TOCL)*, 7(3):499–562, 2006.
85. Nicola Leone, Pasquale Rullo, and Francesco Scarcello. Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation. *Information and computation*, 135(2):69–112, 1997.
86. Beth Levin. *English verb classes and alternations: A preliminary investigation*. University of Chicago press, 1993.
87. Omer Levy and Yoav Goldberg. Dependency-based word embeddings.
88. Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225, 2015.
89. Vladimir Lifschitz, David Pearce, and Agustín Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic (TOCL)*, 2(4):526–541, 2001.
90. Vladimir Lifschitz and Hudson Turner. Splitting a logic program. In *ICLP*, volume 94, pages 23–37, 1994.
91. Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.
92. John W Lloyd. *Foundations of logic programming*. Springer Science & Business Media, 2012.

93. Jorge Lobo, Jack Minker, and Arcot Rajasekar. *Foundations of disjunctive logic programming*. MIT press, 1992.
94. Kevin Lund and Curt Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208, 1996.
95. Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. Yago3: A knowledge base from multilingual wikipedias. In *7th Biennial Conference on Innovative Data Systems Research*. CIDR Conference, 2014.
96. Alexis Manaster-Ramer. Dutch as a formal language. *Linguistics and Philosophy*, 10(2):221–246, 1987.
97. Wiktor Marek and Mirosław Truszczyński. Autoepistemic logic. *Journal of the ACM (JACM)*, 38(3):587–618, 1991.
98. Hiroshi Maruyama. Structural disambiguation with constraint propagation. In *Proceedings of the 28th annual meeting on Association for Computational Linguistics*, pages 31–38. Association for Computational Linguistics, 1990.
99. Ryan T McDonald and Joakim Nivre. Characterizing the errors of data-driven dependency parsing models. In *EMNLP-CoNLL*, pages 122–131, 2007.
100. Ryan T McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al. Universal dependency annotation for multilingual parsing. In *ACL (2)*, pages 92–97. Citeseer, 2013.
101. Oren Melamud, Omer Levy, Ido Dagan, and Israel Ramat-Gan. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, 2015.
102. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
103. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
104. George A Miller. Five papers on wordnet. *Technical Report CLS-Rep-43, Cognitive Science Laboratory, Princeton University*, 1993.
105. George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244, 1990.
106. George A Miller and Christiane Fellbaum. Semantic networks of english. *Cognition*, 41(1):197–229, 1991.
107. Jack Minker. On indefinite databases and the closed world assumption. In *International Conference on Automated Deduction*, pages 292–308. Springer, 1982.
108. Jack Minker. Overview of disjunctive logic programming. *Annals of Mathematics and Artificial Intelligence*, 12(1-2):1–24, 1994.
109. Dan I Moldovan, Sanda M Harabagiu, Marius Paşca, Rada Mihalcea, Richard A Goodrum, Corina R Gîrju, and Vasile Rus. Lasso: A tool for surfing the answer net. 1999.
110. Richard Montague. English as a formal language. 1970.
111. Richard Montague. Universal grammar. *Theoria*, 36(3):373–398, 1970.
112. Richard Montague. The proper treatment of quantification in ordinary english. In *Approaches to natural language*, pages 221–242. Springer, 1973.

113. Roberto Navigli and Simone Paolo Ponzetto. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250, 2012.
114. Ian Niles and Adam Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*, pages 2–9. ACM, 2001.
115. Ian Niles and Adam Pease. Mapping wordnet to the sumo ontology. In *Proceedings of the ieee international knowledge engineering conference*, pages 23–26, 2003.
116. Joakim Nivre. *Inductive dependency parsing*. Springer, 2006.
117. Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106, 2005.
118. Barbara BH Partee, Alice G ter Meulen, and Robert Wall. *Mathematical methods in linguistics*, volume 30. Springer Science & Business Media, 2012.
119. Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. *arXiv preprint arXiv:1508.00305*, 2015.
120. Panupong Pasupat and Percy Liang. Inferring logical forms from denotations. *arXiv preprint arXiv:1606.06900*, 2016.
121. Carl Pollard and Ivan A Sag. *Head-driven phrase structure grammar*. University of Chicago Press, 1994.
122. John M Prager. Open-domain question-answering. *Foundations and trends in information retrieval*, 1(2):91–231, 2006.
123. Teodor C Przymusinski. Static semantics for normal and disjunctive logic programs. *Annals of Mathematics and Artificial intelligence*, 14(2-4):323–357, 1995.
124. Geoffrey K Pullum and Gerald Gazdar. Natural languages and context-free languages. *Linguistics and Philosophy*, 4(4):471–504, 1982.
125. James Pustejovsky. The generative lexicon. *Computational linguistics*, 17(4):409–441, 1991.
126. James Pustejovsky. Type coercion and lexical selection. In *Semantics and the Lexicon*, pages 73–94. Springer, 1993.
127. James Pustejovsky and Pierrette Bouillon. Aspectual coercion and logical polysemy. *Journal of semantics*, 12(2):133–162, 1995.
128. Willard Van Orman Quine, Patricia Smith Churchland, and Dagfinn Føllesdal. *Word and object*. MIT press, 2013.
129. Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140, 2016.
130. Josef Ruppenhofer, Michael Ellsworth, Miriam RL Petruck, Christopher R Johnson, and Jan Scheffczyk. Framenet ii: Extended theory and practice, 2006.
131. Giorgio Satta. Tree-adjoining grammar parsing and boolean matrix multiplication. *Computational linguistics*, 20(2):173–191, 1994.
132. Ingo Schröder. Natural language parsing with graded constraints. *Unpublished PhD Thesis, University of Hamburg, Hamburg*, 2002.
133. Sebastian Schuster and Christopher D Manning. Enhanced english universal dependencies: An improved representation for natural language understanding tasks. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, 2016.

134. Hinrich Schütze. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123, 1998.
135. Petr Sgall, Eva Hajicová, and Jarmila Panevová. *The meaning of the sentence in its semantic and pragmatic aspects*. Springer Science & Business Media, 1986.
136. Libin Shen, Anoop Sarkar, and Aravind K Joshi. Using ltag based features in parse reranking. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 89–96. Association for Computational Linguistics, 2003.
137. Stuart M Shieber. Evidence against the context-freeness of natural language. In *The Formal complexity of natural language*, pages 320–334. Springer, 1985.
138. Push Singh, Thomas Lin, Erik T Mueller, Grace Lim, Travell Perkins, and Wan Li Zhu. Open mind common sense: Knowledge acquisition from the general public. In *OTM Confederated International Conferences On the Move to Meaningful Internet Systems*, pages 1223–1237. Springer, 2002.
139. Robert Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. *arXiv preprint arXiv:1612.03975*, 2016.
140. Mark Steedman. *The syntactic process*, volume 24. MIT Press, 2000.
141. Parsons Terence. Events in the semantics of english: A study in subatomic semantics, 1990.
142. Lucien Tesnière. *Elements of structural syntax*. John Benjamins Publishing Company, 2015.
143. Viet Dinh Tran. *Application of the semantic network ConceptNet*. PhD thesis, 2016.
144. Allen Van Gelder, Kenneth A Ross, and John S Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM (JACM)*, 38(3):619–649, 1991.
145. Krishnamurti Vijay-Shanker and David J Weir. Parsing some constrained grammar formalisms. *Computational Linguistics*, 19(4):591–636, 1993.
146. Morton E Winston, Roger Chaffin, and Douglas Herrmann. A taxonomy of part-whole relations. *Cognitive science*, 11(4):417–444, 1987.
147. Ludwig Wittgenstein. *Philosophical Investigations*. Blackwell, 1953.
148. Mary McGee Wood. *Categorial Grammars (RLE Linguistics B: Grammar)*. Routledge, 2014.
149. William A Woods. Semantics and quantification in natural language question answering. *Advances in computers*, 17:1–87, 1978.
150. Dell Zhang and Wee Sun Lee. Question classification using support vector machines. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32. ACM, 2003.
151. Dell Zhang and Wee Sun Lee. A web-based question answering system. 2003.