Comparirà in: *Bianchi, Cadei, Chesi, Lazzari, Maffei (2009) Informatica Umanistica. McGraw Hill*

4.0 Lo studio della lingua e la rappresentazione dell'informazione linguistica nell'era digitale

Guardando un film di fantascienza è abbastanza comune imbattersi in una "macchina" che interagisce in linguaggio naturale con gli esseri umani, cioè parla fluentemente, comprende l'eloquio di ogni interlocutore umano (magari padroneggiando adeguatamente lingue diverse) e risponde sensatamente a domande che le vengono fatte sui temi più svariati. Il caso più interessante (spesso citato in testi che parlano di *Intelligenza Artificiale* ed in particolare del *Processamento Automatico del Linguaggio*, Jurafsky e Martin 2008) è quello di HAL 9000: nel film di Kubrick, 2001 Odissea nello Spazio, il computer di bordo dell'astronave che conduce gli umani verso l'esplorazione dello spazio sorprende gli spettatori riuscendo a dialogare fluidamente su qualsiasi tema con gli astronauti. Per la realizzazione del film, che uscì nel 1968, Kubrick (il regista) e Clark (lo sceneggiatore del film) consultarono i migliori ricercatori di università (MIT, Harvard, Berkley) e centri di ricerca (NASA, Bell Laboratories) americani e questo permise loro di costruire questa suggestiva visione del futuro di lì ad un quarto di secolo. In questo senso, il film sintetizza in un modo sociologicamente interessante alcune delle principali aspettative delle più brillanti menti che lavoravano su questi temi e fornisce una previsione sul momento in cui queste ricerche avrebbero dato i loro frutti più maturi.

Adesso che il nuovo millennio è arrivato, sorridiamo all'ipotesi di Hall 9000 (Stork 1997) visto che neppure il correttore ortografico del nostro Word Processor soddisfa appieno le nostre aspettative e (quasi) nessuno parla con il proprio PC con la speranza di essere ascoltato o di ricevere risposte sensate. Le ragioni di questo "fallimento" sono in parte spiegate in questo capitolo, ma anche, e soprattutto, si cercherà di creare un percorso attraverso quello che di buono è stato fatto nel campo del processamento automatico del linguaggio con l'avvento dell'era digitale. Google, ad esempio, non era stato previsto nel 1968, ma oggi, questo sofisticato strumento di processamento di informazioni linguistiche, ci rende quotidianamente la vita più facile. Per capire, ad esempio, com'è possibile ottenere risultati interessanti da un motore di ricerca, cercheremo di convincere il lettore che è necessario un studio sistematico della nostra lingua e del modo di rappresentarla digitalmente utilizzando strategie efficienti ed efficaci rispetto ai fini che ci prefiggiamo e alla mole di informazioni da processare.

In questa prospettiva, la prima sezione di questo capitolo è pensata per fornire le informazioni fondamentali sulle proprietà che caratterizzano le lingue naturali, ad esempio la *ricorsività* e il concetto di *struttura sintagmatica* (fondamentali per descrivere sia i linguaggi artificiali che le lingue naturali), e altre, quali l'*ambiguità*, che spesso non vogliamo includere nei linguaggi che usiamo per comunicare con i calcolatori.

Nella seconda sezione, verrà spiegato come si rappresenta esplicitamente un linguaggio: quali sono le componenti di cui abbiamo bisogno per veicolare un messaggio semanticamente sensato e come questo senso venga costruito *composizionalmente*. Esploreremo in questa sessione anche le banche dati che custodiscono le informazioni linguistiche (i *corpora*) e come le proprietà linguistiche specificate nella prima sezione possano venire codificate in questi archivi di conoscenza linguistica.

Nella terza sezione saremo in grado di comprendere perché HAL 9000 (ancora) non esiste: in particolare cercheremo di capire quali sono le proprietà linguistiche che riusciamo a gestire automaticamente con successo e quali meno ed il perché di questa difficoltà. Analizzeremo gli strumenti per interrogare i corpora

descritti nella sezione precedente, quelli per estrarre indici di frequenza, lessici e grammatiche. Infine torneremo sui motori di ricerca (rif. Capitolo 3.4) e su come funziona da un punto di vista squisitamente linguistico il processo di Information Retrieval (vedere Capitolo 6.0 per un approfondimento di questo concetto applicato alla ricerca bibliografica).

4.1. Lingue naturali e linguaggi artificiali

Le macchine non "parlano" come gli esseri umani.

Questa affermazione non dovrebbe sorprendere la maggior parte dei lettori visto che:

a. è abbastanza evidente che alle più banali interrogazioni in linguaggio naturale i nostri PC non reagiscono come vorremmo;
b. non è difficile constatare che i computer "comunicano" tra di loro e con noi, usando un linguaggio che non è propriamente "umano" ma che comunque condivide qualcosa con ciò che intuitivamente definiamo una *lingua naturale*.

Prima di tornare sui concetti chiave di *comunicazione*, *lingua naturale* e *linguaggio artificiale* (rif. Capitolo 1.1), facciamo un piccolo esperimento: proviamo a digitare sul nostro motore di ricerca preferito domande del tipo "cosa danno stasera al cinema?" o "chi era il protagonista dell'ultimo film di Moretti?" a cui qualsiasi nostro amico che abbia a disposizione l'informazione rilevante riuscirebbe a rispondere senza fatica (integrando ad esempio informazioni fondamentali che sono assenti nel testo della nostra domanda quali la città in cui ci interessa andare al cinema o il film che corrisponde alle caratteristiche elencate). Osservando il risultato della nostra ricerca (*query*) dovremmo notare almeno due cose:

(1) a domande puntuali, raramente un motore di ricerca ci risponderà con risposte puntuali;
(2) per ottenere risposte utili dovremmo "raffinare" la nostra ricerca, utilizzando ad esempio parole chiave come "cinema Pisa" oppure " protagonista caimano Moretti".

[Information Retrieval Vs. Information Extraction] La prima osservazione ci permette di sottolineare che il processo di risposta a domande puntuali, che tecnicamente si chiama *estrazione dell'informazione* (*Information Extraction, IE*), prevede un'analisi di un insieme di testi e la rielaborazione del contenuto di parte di questi al fine di proporre all'utente una risposta alla sua domanda che sia il più concisa, corretta e rilevante possibile, magari recuperando e/o inferendo informazioni utili da documenti diversi. Questo processo è sostanzialmente distinto da quello di *recupero dell'informazione* (*Information Retrieval, IR*) che la maggior parte dei motori di ricerca cercano di implementare: l'IR consiste essenzialmente nella selezione di documenti rilevanti, recuperati da una collezione finita di testi, in base a dei precisi parametri linguistici (i.e. parole chiave). In questo capitolo ci occuperemo essenzialmente del secondo tipo di task e di cosa occorre sapere sulla lingua che parliamo per rappresentare e risolvere il problema dell'IR ma non solo: una riflessione sistematica sul linguaggio ci permette di archiviare in modo più efficiente l'informazione semantica, documentare in modo migliore le lingue umane e capire le riflessioni che sono alla base dei compiti di processamento automatico del linguaggio (Natural Language Processing, NLP) che vanno dalla correzione ortografica (spell/grammatical checking) alla traduzione automatica (Machine Translation, MT), passando per il riconoscimento/sintesi del parlato (Speech Recognition/Synthesis).

[Proprietà caratteristiche di lingue e linguaggi] La seconda osservazione invece ci permette di partire dalla considerazione intuitiva che cose come "cinema Pisa" o "protagonista caimano Moretti" non sono esattamente espressioni "naturali", per iniziare a definire quelle che invece sono le proprietà fondamentali delle nostre lingue naturali: prima di tutto osserviamo che entrambe le espressioni "chi era il protagonista dell'ultimo film di Moretti?" e "protagonista caimano Moretti" cercano di isolare un contenuto semantico preciso, cioè *denotare* una particolare entità, che dall'espressione linguistica prodotta e dalla conoscenza che abbiamo del mondo, si inferisce essere animata, anzi, umana (più esattamente un attore) che in particolare ha recitato in un certo film diretto da un preciso regista. Questa informazione semantica nella prima frase è vincolata precisamente dalla giustapposizione delle parole che vengono interpretate in precisi

gruppi, detti *costituenti* (vedi 4.1.2), e che contribuiscono, attraverso la loro relativa posizione, a determinare la composizione del significato semantico. Nella seconda espressione, il contenuto è determinato *intersettivamente*, cioè, parlando in termini insiemistici, una buona "risposta" alla "domanda" posta dovrebbe identificare un'entità che ha contemporaneamente le proprietà di essere "protagonista", "caimano" e "moretti"; nel più semplice dei mondi possibili, questa "entità" è un documento testuale in cui queste tre parole compaiono.

Per chiarire adeguatamente questo punto cruciale, cerchiamo di sottolineare alcune proprietà rilevanti che distinguono le due espressioni:

Proprietà	***Espressione A*** *"chi era il protagonista dell'ultimo film di Moretti?"*	***Espressione B*** *"protagonista caimano moretti"*
Ordine delle parole	cruciale (e.g. “era il chi protagonista Moretti dell' film ultimo di” non ha alcun senso)	irrilevante (e.g. “caimano protagonista moretti” ha lo stesso significato di "protagonista caimano moretti")
Uso elementi privi di contenuto referenziale	“era” esprime tempo e accordo, “il” indica la definitezza del nome a cui si riferisce, “ultimo” modifica una proprietà di “film”, “di” introduce il regista del film	assenti in linea di massima; operatori booleani quali la congiunzione (AND), la disgiunzione (OR) e la negazione (NOT) possono essere comunque presenti in certi contesti.
Ambiguità	“era” in italiano standard è sia verbo (Marco *era* appena uscito) che sostantivo (*era* geologica)	“moretti” può essere sia nome di persona (Nanni *Moretti*) che aggettivo (quei ragazzini *moretti*)

[Elementi funzionali] Inizia quindi a delinearsi una certa opposizione ragionevolmente discriminabile tra il linguaggio naturale (Espressione A) da una parte, con la sua ricca articolazione di elementi funzionali, cioè quelle particelle linguistiche quali articoli, preposizioni e flessioni di accordo che definiscono l'impalcatura su cui le parole contenuto (nomi, verbi e aggettivi) si posizionano e che aiutano a determinarne il significato, ed un linguaggio "semplificato" (Espressione B) dall'altra, in cui gli elementi funzionali che si possono utilizzare sono decisamente in numero minore, inoltre l'ordine delle parole pare in larga misura irrilevante.

Chiariamo subito che ovviamente esistono diversi tipi di “Espressioni B” e che questo particolare linguaggio “semplificato” o “controllato” permette interazioni più produttive, ad esempio, con i motori di ricerca. Di certo, non può essere ancora definito “artificiale” in senso stretto (rif. Capitolo 1.2), poiché ancora conserva una proprietà che caratterizza le lingue naturali, ma che fa letteralmente impazzire i calcolatori: l'*ambiguità*.

[Ambiguità] Per capire cosa significa avere a che fare con l’ambiguità, basta cercare la parola "cuore" su di un motore di ricerca: i documenti che vi verranno restituiti parleranno sia di infarti che di delusioni amorose. Questo perché la parola “cuore”, in italiano, è *ambigua* ovvero almeno due significati sono adeguatamente associabili al lemma “cuore”: “organo” (un attacco di cuore) e “relazione amorosa” (cuore infranto). Se non si aggiungono altre parole chiave la query sarà sempre ambigua e il motore di ricerca non potrà che restituirvi entrambe le accezioni del termine.

Siamo quindi di fronte all'ambiguità (nel caso di "cuore" si parla di *ambiguità semantica*, concetto su cui torneremo nel paragrafo 4.1.2) quando le informazioni linguistiche che ci vengono fornite non ci permettono di selezionare una sola interpretazione possibile. L'ambiguità è una proprietà pervasiva a livello linguistico e fondamentale per l'espressività umana. Se non ci fosse l'ambiguità moltissime poesie e opere di prosa sarebbero aride, risolvere le parole crociate sarebbe noiosissimo e i fratelli Marx non avrebbero mai avuto successo. Per essere sicuri di venir compresi dal computer, però, un linguaggio deve essere completamente esplicito e non ambiguo (o almeno ambiguo in un modo controllabile).

Avendo a disposizione un numero finito di messaggi sufficientemente distinti, diciamo 3, potremmo in effetti associare univocamente ad ognuno di questi un significato specifico: ad esempio ai messaggi "apri file", "salva file", "chiudi file" potremmo associare azioni che il calcolatore riesce a fare (quali appunto aprire, salvare e chiudere un file) definendo così la "semantica" dei messaggi. Ma questo non ci basta a rappresentare digitalmente la lingua umana. La ragione è molto semplice: il numero di frasi che ogni lingua naturale permette di produrre è infinito. Inoltre alla stessa frase, cioè esattamente alla stessa sequenza di parole (es. "Gianni ha visto Maria con il cannocchiale"), possono essere associati significati distinti ("Maria aveva il cannocchiale" oppure "Gianni usava un cannocchiale per vedere Maria").

Nel prossimo paragrafo (4.1.1) affronteremo il problema del numero infinito di messaggi dotati di significato che ogni lingua naturale permette di esprimere ed in quello successivo (4.1.2) parleremo ancora di ambiguità.

4.1.1 L' uso infinito di mezzi finiti

Probabilmente questa è la prima volta che leggete una frase un po' bizzarra come la seguente:

(1) La caduta dei gravi è un'incombenza estremamente pesante

Nonostante la stranezza "semantica" percepita, un parlante nativo dell'italiano non avrà nessuna difficoltà a comprendere quest'espressione composta dalla giustapposizione completamente originale di parole del dizionario italiano. L'originalità della frase non è ovviamente legata ad un'ipotetica creatività dello scrittore; in effetti di frasi che non avete mai letto né sentito, non ce ne sono solo qualche decina, ma un numero infinito ed è piuttosto semplice rendersene conto:

(2) Penso che sia un'incombenza estremamente pesante la caduta dei gravi
(3) Gianni pensa che io pensi che […] che sia un'incombenza estremamente pesante la caduta dei gravi
(4) Penso che sia un'incombenza estremamente pesante la caduta dei gravi che sono stati studiati da Galileo che […] che […]
(5) Penso che sia un'incombenza estremamente pesante la caduta dei gravi e il moto dei pianeti
(6) Penso che sia un'incombenza decisamente pesante la caduta dei gravi e il moto dei pianeti

Potremmo continuare all'infinito, complementando una frase con un'altra (e.g. (2), (3)), espandendo con frasi relative i sintagmi nominali (e.g. (4), ("E venne l'acqua che spense il fuoco che bruciò il bastone che picchiò il cane che morse il gatto che si mangiò il topo che al mercato mio padre comprò" Branduardi 1976), congiungendoli tra di loro in blocchi di due (o tre, o quattro...) elementi (e.g. (5)), rimpiazzando un modificatore con un altro (e.g. (6)). Le possibilità sono certo molte, anche se la tipologia delle "regole/trasformazioni" che si possono applicare non è infinita:

(7) * Che penso incombenza un' sia estremamente pesante e dei caduta gravi il dei pianeti moto

La frase in (7) è *agrammaticale*, e si usa segnalarla con un segno * all'inizio per indicarne questo status, cioè nessun parlante nativo la riconoscerà mai come frase ben formata appartenente alla lingua italiana.

[Grammaticalità] Facciamo attenzione: *agrammaticale* non vuol dire semplicemente "strana": esistono frasi semanticamente strane che possono però essere tranquillamente prodotte e comprese da un parlante nativo, esattamente come la frase in (1) o come il famoso esempio di Chomsky (1957):

(8) Idee verdi senza colore dormono furiosamente (originale: “colorless green ideas sleep furiously”)

La *struttura* della frase ci permette di interpretare correttamente quest'espressione sebbene il contenuto semantico risulti abbastanza bislacco. Il nodo chiave è in effetti proprio questo concetto di *struttura*, che ci permette di combinare in un modo preciso e semanticamente significativo elementi lessicali. Rimandiamo al paragrafo 4.1.1.2 la discussione più approfondita di questo concetto, ma vale la pena soffermarci adesso sul fatto che lo studio sistematico della lingua prevede esattamente l'identificazione di una procedura completamente esplicita per isolare in modo preciso le espressioni che sono ben-formate, cioè *grammaticali*, da quelle che non lo sono e che quindi non appartengono alla lingua.

[Gestire l’infinito] Visti gli esempi in (2)-(6) dovremmo quindi aspettarci una procedura che non elenchi semplicemente tutte le possibili frasi in una lingua: visto che queste frasi sono infinite, tale lista esaustiva sarebbe infinita e perciò impossibile da memorizzare, sia nel nostro cervello che su calcolatore (in effetti, un tale sforzo di memorizzazione richiederebbe un tempo infinito anche solo per passare in rassegna una sola volta ogni frase e una memoria altrettanto infinita per archiviare l’informazione passata in rassegna).

Cosa importante, però, è che il concetto di infinito di per sé non è ciò che spaventa: pensate ai numeri naturali ed in genere all'abilità di contare. Si fa uso di un “concetto”, quello di *successore*, per produrre/comprendere un insieme infinito di oggetti (i numeri appunto) senza memorizzarli tutti, semplicemente partendo da un insieme finito di simboli (dieci caratteri numerici, nel nostro sistema decimale): dato un numero iniziale, lo zero, che non è successore di nessun altro numero, e il concetto di unità, il numero 1, ogni numero naturale *n*, es. 1029933, avrà sempre un *successore*, cioè il numero *n+1*, i.e. 1029934, che sarà a sua volta un numero naturale. Questo è un postulato (più precisamente un *assioma*, "l'assioma del successore") che, formalizzato precisamente, è fondamentale per tutta la matematica.

In modo molto simile, un’idea di “successore” è presente anche nella concezione (formale) di linguaggio: la componente finita (al netto di eventuali e sporadiche aggiunte lessicali quali neologismi o derive morfologiche) è composta dal lessico e dalle regole di combinazione degli elementi lessicali. Regole che determinano meccanicamente (cioè senza nessuna ambiguità) se una frase è producibile in una determinata lingua, cioè permettono di dire che (1)-(6) e (8) sono frasi producibili in italiano mentre (7) non lo è.

Un tipico esempio di siffatte regole è la *regola di concatenazione* che può essere espressa (informalmente) come segue:

(9) **Regola di concatenazione** (definizione informale)
se *X* e *Y* sono espressioni grammaticali nella nostra lingua *L*, allora anche *XY* è un'espressione grammaticale

Ad esempio se "Mario mangia " (la nostra espressione "*X*") e "una mela " (la nostra espressione "*Y*") sono espressioni che appartengono (in termini insiemistici la relazione di appartenenza si esprime con il segno "$\in$") alla lingua in esame (chiamiamola *L*) (cioè, in termini un po' più precisi: $X, Y \in L$), allora anche "Mario

mangia una mela" appartiene alla lingua italiana (*XY* $\in$ *L*). Ma allora se *XY* appartengono a *L* possono essere considerate esattamente come sostituibili ad *X* e quindi venire (ri)combinate con *Y* formando *XYY*. *L* a questo punto assomiglierà sempre meno all'italiano che conosciamo (la frase "Mario mangia una mela una mela" non è in effetti grammaticale in italiano), ma questo non invalida il nostro ragionamento; *L* sarà un linguaggio arbitrario che con una sola regola e due soli elementi, riesce a produrre un numero infinito di frasi:

(10)**Regola di concatenazione** (definizione formale)

$X, Y \in L \Rightarrow XY \in L$

Il simbolo "$\Rightarrow$" indica l'implicazione logica "se … allora"; la regola si legge quindi "se *X* e *Y* appartengono a *L*, allora anche *XY* appartiene a *L*".

Ecco quindi spiegato il trucco dell'uso infinito di mezzi finiti: la (ri)combinazione ricorsiva degli stessi elementi, cioè l'iterazione di una regola che volta per volta estende l'espressione grammaticale aggiungendo un elemento noto; questo produce sempre una nuova e originale espressione che, per definizione, è grammaticale (visto che è generata da una regola che è parte della nostra grammatica).

Per creare effettivamente una *grammatica* per l'italiano dobbiamo costruire regole simili a quella in (10) ma un po' più restrittive: un classico esempio per rappresentare una grammatica (che in questo senso viene definita *generativa*, cioè completamente esplicita, al punto che specificandola si conosce una procedura effettiva per sapere esattamente quali frasi faranno parte di una lingua e quali no) sono le *regole di riscrittura* (Chomsky 1957). Prima di iniziare a comprendere la logica che soggiace a queste regole (4.1.1.3) è necessario riflettere con maggiore chiarezza su cosa si intende per grammatica generativa (4.1.1.1) e per descrizione strutturale (4.1.1.2).

4.1.1.1. La grammatica generativa

Una *grammatica generativa* (o *formale*) per un linguaggio *L* è un insieme finito di *istruzioni/regole* che permettono di *generare/riconoscere* tutte e sole le frasi appartenenti a *L* e(d eventualmente) di assegnare a queste frasi un'adeguata *descrizione strutturale* (paragrafo 4.1.1.2).

Chiariamo alcuni concetti fondamentali che ci permettono di descrivere un linguaggio artificiale, ma potenzialmente anche una lingua naturale: l'*alfabeto*, il *vocabolario*, il *linguaggio*, nozioni queste, che sono propedeutiche al concetto formale di *grammatica*.

[Alfabeto] L'*Alfabeto* (chiamiamolo *A*) è un insieme finito di caratteri.

Solitamente gli insiemi finiti si esprimono estensionalmente, cioè fornendo una lista esaustiva degli elementi che li compongono; per fare questo si usano le parentesi graffe e si separano i singoli elementi usando virgole:

(11) A = {a, s, c, l}

Vale la pena sottolineare che un insieme così definito non specifica nessun ordine tra i suoi elementi: in questo senso gli insiemi A_1 = {s, c, a, l} e A_2 = {l, a, s, c} sono entrambi equivalenti ad A (un insieme ordinato di elementi, solitamente, si esprime sostituendo le parentesi graffe con quelle angolate: A_1 = <s, c, a, l> e A_2 = <l, a, s, c> così definiti sono insiemi diversi).

Gli elementi dell'alfabeto possono venire combinati in *parole* utilizzando l'operazione di concatenazione descritta nel paragrafo precedente. Potenzialmente il numero di parole generabili da un dizionario finito è infinito, come discusso precedentemente ed in particolare, l'insieme di tutte le stringhe possibili costruite concatenando elementi di A si definisce A* (si legge "A star" dove il simbolo * non indica l'agrammaticalità della frase, come abbiamo visto precedentemente, ma è un operatore logico che esprime esattamente l'applicazione ricorsiva di ogni possibile concatenazione compresa quella nulla; giusto per riferimento, tale operatore è detto *chiusura di Kleene*, dal nome del matematico, Stephen Cole Kleene, che lo ha definito).

[Vocabolario] Volendo riflettere su una lingua naturale, dato un alfabeto, solo alcune combinazioni di lettere saranno effettivamente parole della nostra lingua. In questo senso potremmo definire il *Vocabolario* (V) come un insieme (potenzialmente in)finito di parole costruite concatenando elementi di A ($V \subseteq A^*$, cioè il vocabolario V è incluso nell'insieme di tutte le possibili concatenazioni di caratteri dell'alfabeto A).

Ad esempio:

(12) V = {ala, cala, casa, la, sala, scala}

(12) è un vocabolario finito (composto da 6 parole). **[Linguaggio]** Queste parole possono essere a loro volta combinate in espressioni di più parole, ad esempio "la casa". Otterremo così un *Linguaggio* (*L*): un nuovo insieme (potenzialmente in)finito di frasi, costruite concatenando elementi di V ($L \subseteq V^*$).

A questo punto resta da capire come limitare questa abilità generativa, visto che cose come "la la la casa la" o "casa sala scala la" non sono espressioni che vorremmo includere nel nostro frammento di lingua italiana ma che vengono generate dalla nostra grammatica se vi includiamo la regola di concatenazione incondizionatamente. Come dichiarato ad inizio paragrafo, una grammatica formale deve permetterci di fare esattamente questo: esprimere "un insieme di regole che permettono di *generare/riconoscere* tutte e sole le frasi appartenenti a *L* e(d eventualmente) di assegnare a queste frasi un'adeguata descrizione strutturale". È a questo punto fondamentale chiarire che cosa si intende per *struttura*.

4.1.1.2. La descrizione strutturale di una frase

[Costituenti sintattici] Il concetto di struttura si basa sull'idea di *costituente sintattico*, intuitivamente introdotto in 4.1.1: un *costituente* è un'entità potenzialmente composta da più parole (ad esempio in (13).a alcuni costituenti sono indicati dalle parentesi quadre) che si comporta uniformemente rispetto a determinati test quali, ad esempio, la *pronominalizzazione* (13).b, il *movimento* (ovvero il riposizionamento in punti diversi di un gruppo di parole all'interno della stessa frase) (13).c'-c'' e la *costruzione di frasi scisse* (14) (indici uguali indicano che il costituente indicizzato e la posizione di default espressa in (13).a, indicata con il simbolo "_" nelle frasi successive, si riferiscono allo stesso oggetto):

(13) a. Devo mangiare [il pesce] [dopo domani]
b. [lo] $_i$ devo mangiare _ $_i$ [dopo domani]
b'. * [lo] $_i$ devo mangiare [il _ $_i$] [dopo domani]
c. [Cosa] $_i$ devo mangiare _ $_i$ [dopo domani]?
c'. [Il pesce] $_i$ devo mangiare _ $_i$ dopo domani!
c''. *[Cosa] devo mangiare [il _ $_i$] dopo domani?

(14) a. È [il pesce] $_i$ che devo mangiare _ [dopo domani]
b. *È [il pesce dopo] che devo mangiare _ $_i$ domani

Un test sintattico come la pronominalizzazione (13).b ci dice che solo certi tipi di costituenti posso essere rimpiazzati da un pronome: un sintagma nominale comprensivo di articolo può essere pronominalizzato ((13).b) ma non il solo nome, lasciando l'articolo pendente nella posizione originale (notare in questo senso l'agrammaticalità di (13).b'); esattamente la stessa riflessione vale per l'elemento interrogativo "cosa" che non può sostituire solo il nome lasciando l'articolo nella posizione di base ((13).c Vs. (13).c'). Infine la frase scissa ci dice che il cluster "articolo nome" è effettivamente un costituente ((14).a) e che se cerchiamo di includervi un elemento adiacente che però fa parte di un altro costituente (ad esempio l'avverbio "dopo") otteniamo un'espressione agrammaticale ((14).b). È fondamentale osservare che tali test discriminano perfettamente quelli che sono i veri costituenti ((13).a,b,c,c', (14).a) da quelli che non lo sono ((13).b',c'', (14).b).

Si può quindi constatare che i costituenti sono delle classi naturali di oggetti, ed in effetti, solo i costituenti dello stesso tipo possono essere sostituiti ((15).a Vs. (15).b) e/o coordinati ((16).a Vs. (16).b).

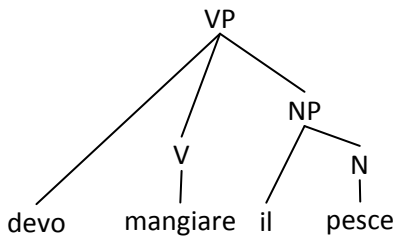
(15) a. Devo mangiare [la carne] dopo domani
b. *Devo mangiare [di ferro] dopo domani
(16) a. Devo mangiare [[il pesce] e [l'insalata]] dopo domani
b. *Devo mangiare [[il pesce] e [prima dell'una]] dopo domani

Intuiamo così la necessità di introdurre nel nostro vocabolario una serie di elementi che non sono esattamente parole, ma che ci permettono di indicare univocamente certe tipologie di costituenti e di elementi in modo da esprimere precisamente i vincoli strutturali sensibili a certe proprietà quali la sostituibilità, la pronominalizzazione, il movimento o la coordinazione ed in genere la distribuzione lineare delle parole all'interno di una frase.

[Elementi non terminali] Introdurremo quindi nel nostro vocabolario anche termini quali *Nome* (*N*), *Verbo* (*V*) e rispettivamente gli indicatori dei costituenti nominali (*Sintagma Nominale*, *SN*, o *Nominal Phrase*, *NP*) e dei costituenti verbali (*Sintagma Verbale*, *SV*, o *Verbal Phrase*, *VP*) che saranno le "proiezioni" degli elementi nominali e verbali.

In questo caso i nostri sintagmi potranno essere rappresentati come segue usando una rappresentazione con parentesi quadre ((17).a, più compatta) o l'equivalente albero sintattico ((17).b, visivamente più esplicita delle relazioni tra i nodi dell'albero e le varie parole):

(17) a. $[_{VP}$ Devo $[_{V}$ mangiare$]$ $[_{NP}$ il $[_{N}$ pesce $]]]$
b.



Tutte le parole della frase dovranno avere una "categoria" sintattica, anche quelle che non contribuiscono con un contenuto "referenziale" (nomi) e/o "azionale" (verbi); quindi tutte le particelle "grammaticali" (che, come abbiamo pocanzi detto, si definiscono *funzionali*) quali, ad esempio gli articoli, le preposizioni e gli ausiliari, saranno categorizzare: *Articolo* (*D* come *Determiner*, in inglese), *Preposizione* (*P*), *Ausiliare/Modale* (*Aux/Mod*) etc. e così anche gli *Aggettivi* (*A*), gli *Avverbi* (*ADV*) in modo che ogni parola abbia un singolo

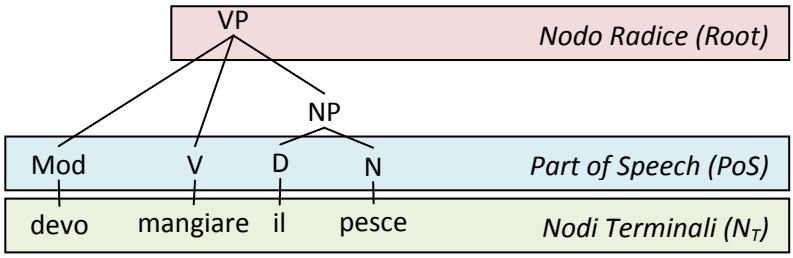
nodo padre che ne specificherà la categoria morfo-sintattica. Questa etichetta si chiama *Part of Speech* (*PoS*).

Le parole saranno sempre le foglie dell'albero sintattico e saranno indicate come elementi *terminali*.

Il nodo più alto (in questo caso la sua etichetta è "VP") è invece unico e viene chiamato nodo *radice*.

Con queste nuove etichette possiamo completare l'albero come indicato:

(17) c.



Questo sforzo descrittivo ci permetterà di circoscrivere la portata di certe regole di concatenazione non a singoli elementi lessicali, ma a classi di elementi lessicali (PoS) e a gruppi strutturati di parole (costituenti). Dicendo ad esempio che un costituente nominale (NP) in Italiano è composto da un articolo (D) e da un nome (N), in questo esatto ordine, riusciamo a generare/riconoscere come espressioni nominali cose come "il pesce", "un cane", "la mela" ecc. ed escludere cose come "pesce dello" o "mela la". Sapendo poi che i sintagmi nominali possono essere argomenti di certi verbi, cioè che un sintagma verbale può essere costituito da un verbo (V) e un sintagma nominale (NP) in questa sequenza, potremmo riconoscere come ben formate frasi del tipo "mangio una mela".

4.1.1.3. Le grammatiche a struttura sintagmatica e le regole di riscrittura

[Grammatiche formali] Lo scopo di una *grammatica formale* è quindi quello di esplicitare in un modo comprensibile per un computer quelli che sono i simboli e le regole che permettono di identificare correttamente l'insieme di espressioni appartenenti ad una determinata lingua e di assegnare a queste espressioni una struttura che esprime alcune proprietà interessanti (quali quella di isolare classi naturali di elementi che sottostanno a determinate condizioni). Una siffatta grammatica deve godere essenzialmente di due proprietà:

- deve essere *esplicita*, ovvero il giudizio di grammaticalità deve essere frutto solo dell'applicazione meccanica delle regole incluse nella grammatica (cioè le frasi grammaticali saranno tutte e sole quelle espressioni che la nostra grammatica potrà generare);
- deve essere *consistente*, ovvero una stessa frase non può risultare allo stesso tempo grammaticale e non grammaticale (cioè una stessa espressione può essere generata o non può essere generata dato un preciso insieme di regole).

[Grammatiche a Struttura Sintagmatica (PSG) e regole di riscrittura] Un noto esempio di grammatiche formali sono le *Grammatiche a Struttura Sintagmatica* (*Phrase Structure Grammar*, *PSG* Chomsky 1965), dove per struttura sintagmatica si intende la descrizione strutturale in (17).

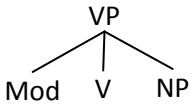
Per capire come funzionano le regole di riscrittura che sono alla base delle grammatiche PSG, si può partire analizzando il nodo radice *VP*: sotto *VP*, l'albero si espande in tre nodi che sono rispettivamente *Mod*, *V* e

NP. Questa fondamentale relazione, definita di *dominanza immediata* tra nodi dell'albero (i.e. VP *domina immediatamente* Mod, VP *domina immediatamente* V e VP *domina immediatamente* NP), oltre all'ordine lineare in cui *Mod*, *V* e *NP* compaiono nella struttura, deve essere resa dalla grammatica e proprio a questo servono le regole di riscrittura, che, per l'espansione in esame si esprimono come segue:

(18) VP → Mod V NP

→ è il simbolo di riscrittura e la regola si legge così: "il nodo non terminale *VP* viene riscritto con la sequenza di nodi *Mod*, *V*, *NP*". Questo corrisponde graficamente al sottoalbero seguente:

(19)



Per generare l'intera descrizione strutturale in (17), dovremmo specificare almeno le seguenti regole:

(20)

Regole di riscrittura non terminali	*Regole di riscrittura terminali*
VP → Mod V NP	Mod → devo
NP → D N	V → mangiare
	D → il
	N → pesce

Può essere utile suddividere in due gruppi, come fatto in (20), le regole in cui il nodo da espandere domina altri nodi che possono essere a loro volta espansi (*regole non terminali*), da quelle in cui il nodo da espandere domina direttamente un elemento terminale (*regole terminali*). Le foglie così identificate compongono il vocabolario terminale di una lingua che, in una grammatica generativa, solitamente si identifica con il simbolo V_T. Se ad esso si aggiunge la lista dei nodi non terminali (V_N) si otterrà il Vocabolario (V) unione di V_T e V_N (in termini insiemistici: $V_T \cup V_N$):

(21)

V_N	V_T
VP	devo
Mod	mangiare
V	il
NP	pesce
D	
N	

L'ultimo elemento che si richiede di specificare in una grammatica a struttura sintagmatica è il nodo radice: un albero risulterà ben formato se solo se al suo vertice ci sarà l'elemento designato come *Root* nella grammatica, che nel nostro caso è *VP* (o, come si indica solitamente, *S* per *Sentence*).

Riassumendo, una grammatica a struttura sintagmatica è una grammatica formale composta da:

- un *Vocabolario* (V) composto da simboli terminali (il lessico, V_t) e non terminali (categorie grammaticali, V_N)
- un insieme di regole di riscrittura (*R*)

- un nodo speciale (appartenente al vocabolario non terminale) che deve essere la radice di ogni albero ben formato (*root*).

Una tale grammatica ci permette di generare una serie di espressioni dotate di struttura semplicemente iniziando ad espandere il nodo radice, continuando ad espandere nodi non terminali, man mano che troviamo regole di riscrittura con tali nodi alla sinistra del simbolo di riscrittura. La generazione si dice *terminata* quando ogni nodo non terminale è espanso in un nodo terminale.

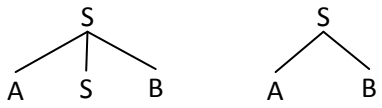
[Un esempio di grammatica PSG] Data una grammatica *G* del tipo descritto, una lingua *L* generata da G (detta *L(G)*), è l'insieme di tutte le stringhe (espansioni terminate) derivabili partendo dal nodo radice. Vediamo un esempio concreto di grammatica giocattolo (toy grammar) e le stringhe/strutture che si riescono a generare:

(22)

V_N	V_T	*Root*	*R*
S	a	S	1. S → A S B
A	b		2. S → A B
B			3. A → a
			4. B → b

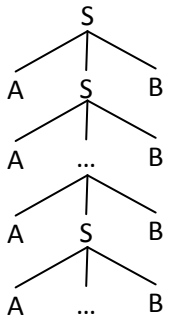
Partiamo con l'espansione di S e notiamo subito un fatto interessante, per quanto apparentemente banale: ci sono due regole (la 1 e la 2) che presentano il nodo S alla sinistra del simbolo di riscrittura. Per come lo abbiamo definito in precedenza, il processo di generazione di stringhe procede ciecamente espandendo tutte le regole che iniziano con un determinato simbolo in esame, quindi dovremmo generare entrambi i sottoalberi seguenti:

(23) 1. 2.



Ipotizziamo però che il compito del nostro computer sia quello di verificare se la grammatica in esame riesce a generare esattamente la stringa "ab"; al primo passo il computer non potrà esserne certo poiché le espansioni in (23) non generano ancora nessuna stringa terminata con "ab" e neppure si potrà dire con certezza, se una delle tre alternative è migliore delle altre senza esplorare ulteriori regole nella grammatica. **[Non-determinismo]** Questa situazione di "incertezza" che richiede un'analisi/espansione ulteriore è definita di *non-determinismo*: il *non-determinismo* è quella condizione in cui una procedura meccanica si trova quando deve scegliere tra varie alternative possibili e tutte risultano ugualmente probabili e legali, poiché non sussistono, nel preciso momento della scelta, informazioni sufficienti che potrebbero farle risparmiare tempo e farle espandere solo le regole che potrebbero portare direttamente alla stringa cercata. Questo concetto di non-determinismo è strettamente legato all'idea di *ambiguità* che verrà considerata nel prossimo paragrafo. Al momento accettiamo il fatto che una procedura di espansione delle regole possa procedere espandendo in parallelo tutte le regole ammissibili. **[Regole ricorsive]** Questo ovviamente potrebbe richiedere molta memoria, e, in alcuni casi, addirittura molta più memoria di quanta un computer possa mai averne: in effetti vale la pena notare un'ulteriore proprietà, stavolta molto meno banale, presente nella prima regola di riscrittura: lo stesso elemento non-terminale (S, per l'appunto) è presente sia alla sinistra che alla destra del simbolo di riscrittura. Reiterando in effetti ciecamente il procedimento di espansione, ci troveremmo nell'imbarazzante situazione di creare una derivazione infinita come la seguente:

(24)



Tele regola, che si definisce *ricorsiva*, se espansa senza controllo non condurrà mai ad una derivazione terminata, indipendentemente da quanto tempo e quanta memoria abbiamo a disposizione.

Il problema non può semplicemente essere risolto eliminando tali regole dalle nostre grammatiche, poiché le regole ricorsive sono fondamentali per ottenere un numero infinito di frasi e questa è una proprietà delle lingue naturali da cui non possiamo prescindere. Se ci precludiamo questa possibilità, in effetti, utilizzeremmo grammatiche che non sono *adeguate* per descrivere i fenomeni naturali che ci interessano. Poiché questo dell'adeguatezza è un concetto importante, torneremo su questo punto nella seconda sezione di questo capitolo. Adesso riflettiamo ancora sulle regole ricorsive e sul modo per limitarle: per bloccare la derivazione in (24) è sufficiente scegliere di sostituire *S* applicando la regola 2 anziché la regola 1 e quindi concludere la derivazione espandendo (regola 3) le *A* con *a* (simbolo terminale) e (regola 4) le *B* con *b* (simbolo terminale). Ecco qua la derivazione completa con menzione delle regole applicate di volta in volta espandendo un nodo non terminale per volta, da sinistra a destra:

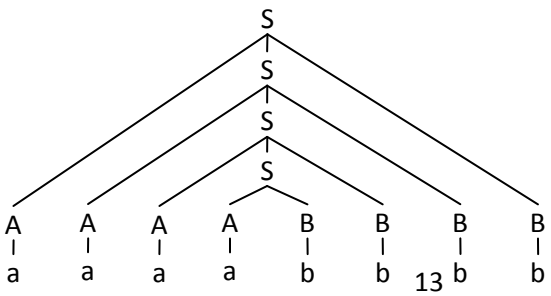
(25)

Stringa prodotta	*Regola applicata*
S	Regola di default (nodo radice)
A S B	Regola 1
A A S B B	Regola 1
A A A S B B B	Regola 1
A A A A B B B B	Regola 2
a A A A B B B B	Regola 3
…	Regola 3
a a a a b B B B	Regola 4
…	Regola 4
a a a a b b b b	Regola 4 (fine derivazione, poiché ogni nodo non-terminale è stato espanso in un nodo terminale)

La struttura è data implicitamente dall'applicazione delle regole ed è la seguente ((26).a e (26).b, come spiegato precedentemente, sono due modi equivalenti per esprimere le relazioni strutturali):

(26) a. $[_S\ [_A\ a\ ]\ [_S\ [_A\ a\ ]\ [_S\ [_A\ a\ ]\ [_S\ [_A\ a\ ]\ [_B\ b\ ]\ ]\ [_B\ b\ ]\ ]\ [_B\ b\ ]\ ]\ [_B\ b\ ]\ ]$

b.



[Dipendenze contabili] Una proprietà interessante che ha il linguaggio generato dalla grammatica descritta in (22) è che il numero di *a* e il numero di *b* è sempre uguale. Questa caratteristica si esprime formalmente usando degli apici che indicano il numero di ripetizioni del termine: ad esempio a^4b^4 corrisponderà alla stringa *aaaabbbb*; nel nostro caso possiamo affermare che ogni stringa generata dalla nostra grammatica è del tipo a^nb^n con *n* numero naturale arbitrariamente grande. La proprietà di mantenere uguale il numero di occorrenze di diversi caratteri all'interno di una stringa è definito *dipendenza contabile* (*counting dependency*). Nel caso citato (a^nb^n) abbiamo una sola *dipendenza contabile*, ma potenzialmente potremo pensare che esistano grammatiche con più dipendenze contabili ($a^nb^mc^nd^m$, con *m* e *n* potenzialmente diversi, oppure $a^nb^nc^n$).

Un importante risultato che i pionieristici studi di Chomsky negli anni '50 hanno ottenuto, è quello di dimostrare che ponendo restrizioni di diverso tipo sulle regole di riscrittura (ad esempio ammettendo non uno soltanto, ma più simboli terminali e/o non-terminali alla sinistra del simbolo di riscrittura) si ottengono grammatiche di potenza generativa diversa, ovvero ci sono grammatiche che possono generare linguaggi (e quindi stringhe e relative descrizioni strutturali) che le altre grammatiche con vincoli diversi non potranno mai generare, perché prive di certi meccanismi di controllo del processo ricorsivo.

Il ragionamento è abbastanza complesso nella sua completezza ed esula dai fini di questo capitolo, ma giusto per avere un'idea di come funzionano le cose, ipotizziamo un semplice vincolo oltre a quello di avere un solo elemento non terminale alla sinistra della regola:

(27) *Vincolo su posizione e numero dei simboli non terminali alla destra del simbolo di riscrittura*
alla destra delle regole di riscrittura può esserci solo un simbolo non-terminale e questo, se presente, deve essere all'estrema destra della parte della regola da riscrivere (cioè alla destra della sequenza di simboli terminali e non-terminali dopo il simbolo di riscrittura)

Le regole ammissibili nelle grammatiche che accettano questo vincolo saranno tutte del tipo (28).a,b,c e non del tipo (28).c,d,e (*X* e *Y* sono elementi non-terminali, *a* e *b* solo elementi terminali):

(28)
a. X → aY
b. X → Y
c. X → X
b. *X → aYb
c. *X → Ya
d. *X → XY

Senza cercare di dimostrare formalmente questa affermazione (per fare questo dovremmo ricorrere alla dimostrazione di un teorema detto *pumping lemma*, Partee et al. 1990), possiamo osservare come le regole ammissibili in una grammatica che accetta il vincolo in (27) non possono generare stringhe con neppure una counting dependency. Per rendersi conto di questo basta osservare che per quanto ci sforziamo, qualsiasi regola cerchiamo di introdurre per sostituire la X o la Y con qualcosa (ad esempio X → aY, Y → b), il numero di elementi terminali che riusciremo a generare sarà solo occasionalmente uguale e mai sistematicamente uguale come nella grammatica in (22). Quindi linguaggi con proprietà tipo a^nb^n non saranno mai generabili da grammatiche come quelle vincolate da (27).

[Dipendenze contabili nelle lingue naturali] Ragionando concretamente sulle lingue naturali, viene da chiedersi cosa c'entri una proprietà come la dipendenza contabile con un'espressione naturale. La risposta la fornisce Chomsky nel suo famoso lavoro del 1957: strutture del tipo "*Se* Gianni pensa che *se* piove *allora*

Maria porta l'ombrello *allora* lui non porta l'impermeabile" esprimono esattamente questa proprietà di dipendenza contabile in quanto una frase S può essere riscritta come "se *S* allora *S*" (*S* → *a S b S*). Tale regola genererà ogni volta espressioni in cui il numero di *se* e *allora* (o di *a* e di *b*) è uguale. Quindi dobbiamo almeno rimuovere il vincolo in (27) per cogliere questa proprietà di cui sembrano godere le lingue naturali.

[Potenza generativa e gerarchia di Chomsky] Ponendo vari vincoli Chomsky dimostra che le grammatiche sono rappresentabili in una gerarchia (concentrica) di inclusione (la *Gerarchia di Chomsky*, Chomsky 1956) in cui le grammatiche che includono le altre (grazie ai vincoli diversi che vengono di volta in volta imposti alle regole di riscrittura), generano linguaggi che non sono generabili dalle grammatiche delle classi inferiori.

Cosa ci interessa sapere è che le grammatiche che accettano un vincolo come quello espresso in (27) si definiscono "regolari" e sono praticamente alla base della gerarchia di Chomsky, mentre le grammatiche che catturano almeno una dipendenza contabile, e quindi proprietà linguisticamente più interessanti, non hanno il vincolo che abbiamo discusso e si identificano con grammatiche che si chiamano *context-free*.

Le grammatiche regolari ci torneranno utili per studiare le *espressioni regolari* (4.3.1.1), ovvero un modo relativamente semplice per isolare un insieme di stringhe che, guarda caso, è proprio generabile dalle grammatiche regolari.

Le grammatiche context-free ci permetteranno invece maggiore facilità nella definizione dei costituenti sintattici e ci permetteranno di annotare in modo utile i dati linguistici su cui vorremmo operare sia per fini documentali che per analisi linguistiche automatiche (4.2.3, 4.3.4.2).

4.1.2. I livelli di rappresentazione della lingua e l'ambiguità

Adesso che abbiamo familiarizzato un po' con grammatiche e strutture sintagmatiche torniamo all'aspetto che suscitava maggiori preoccupazioni per il trattamento automatico delle lingue naturali: l'ambiguità.

[Competenza linguistica] Cominciamo con l'osservare che la nostra *competenza linguistica*, cioè quell'insieme di conoscenze che ci permette di dire con sicurezza se una frase è grammaticale o no e, in caso di ambiguità, di assegnare il corretto significato alle distinte interpretazioni possibili, prevede vari livelli di consapevolezza; cercando di sistematizzare queste tipologie di conoscenze possiamo affermare che un parlante nativo dell'italiano deve sapere che:

a. una parola può iniziare per "*ma…*" (*mare*) ma non per "*mr...*";
b. la e di case ha un valore diverso da quella di mare;
c. "le case sono sulla collina" è una frase ben formata mentre *"case le collina sono sulla" è un'espressione agrammaticale;
d. nella frase "il gatto morde il cane", "morde" esprime l'azione che è svolta dal "gatto" (soggetto grammaticale della frase e agente) e subita dal cane (oggetto grammaticale e paziente dell'azione);
e. "#il tostapane morde il gatto" è un'espressione semanticamente strana (per questo viene indicata con il diacritico "#");
f. l'espressione "la casa" si riferisce ad una casa specifica evidente dal contesto (Vs. "una casa").

Esprimere queste conoscenze in un modo esplicito, significa definire la competenza linguistica: per far ciò, prima di tutto, avremo bisogno di specificare le primitive ad ogni livello (visto che potenzialmente i tratti ed i vincoli sulle combinazioni dei caratteri/suoni, punto a., saranno completamente insensibili ai tratti

semantici e alle loro restrizioni combinatorie, punto e., e/o viceversa). Possiamo quindi parlare di primitive elementari almeno nei seguenti domini:

a. *fonemico* - si esprimono a questo livello le rappresentazioni astratte dei suoni del linguaggio; ad esempio si codificano i fonemi, cioè le più piccole entità acustiche che permettono di discriminare opposizioni significative attraverso tratti distintivi (quali *sordo*/*sonoro* che permettono di distinguere la "s" di "sacco" da quella di "rosa");
b. *morfemico* - si identificano a questo livello i *morfemi*, ovvero i più piccoli raggruppamenti di fonemi dotati di significato (ad esempio, la parola "cani" è composta dai morfemi "can-" radice del nome "cane" ed il morfema "-i" che esprime il plurale maschile);
c. *parole* - è il livello a cui si esprimono i gruppi di morfemi significativi che compongono le parole che si possono trovare nel dizionario, cioè i *lemmi* (anche detti *type*, ad esempio "cane") o le *occorrenze* (anche dette *tokens*, ad esempio "cani");
d. *sintagmatico* - si individuano a questo livello i gruppi tipizzati di parole (sintagma nominale, sintagma verbale etc.) che esprimono i costituenti sintattici e le relazioni di inclusione tra tali costituenti;
e. *tematico/relazionale* - si esprimono i concetti di paziente, agente etc. che specificano le relazioni tematiche e/o modificazionali tra costituenti distinti (ad esempio tra un verbo e il suo complemento oggetto, o tra un nome e l'aggettivo che lo modifica) ;
f. *discorsivo* - si specificano a questo livello concetti quali l'argomento della conversazione (il *topic* del discorso), le relazioni pragmatiche pertinenti (implicazioni/inferenze di vario tipo, elementi focalizzati contrastivamente o introdotti inaspettatamente nel contesto).

Vista la limitatezza di questa trattazione, in questo capitolo ci concentreremo essenzialmente sulle specificazioni del livello morfemico e sintagmatico. Nella prossima sezione (2) faremo solo qualche accenno all'informazione tematico/relazionale e discorsiva. Nella restante parte di questo paragrafo, invece, vorremmo soffermarci sul fatto che se da un lato è fondamentale specificare le primitive ad ogni livello, questo, di per sé, non ci garantisce che tali primitive potranno essere utilizzate correttamente ed efficientemente da un qualsiasi sistema informatico. Per capire questo punto ricorriamo ad un'opposizione classica: *competence* Vs. *performance* (Chomsky 1965) da una parte e *processing* dall'altra.

[Competence Vs. Performance] La distinzione tra *competence* e *performance* permette di isolare produttivamente l'insieme di dati linguistici che ci interessa descrivere: proprietà come la ricorsività infinita e le dipendenze contabili chiaramente non troveranno mai una completa implementazione in un'espressione in lingua naturale visto che, come abbiamo discusso nel paragrafo precedente, ci vorrebbe un tempo infinito per produrre espressioni infinite. L'idea di fondo, che rende comunque questi fenomeni interessanti, è che non produciamo/comprendiamo espressioni con queste proprietà solo perché quando parliamo e cerchiamo di comprendere un'espressione linguistica abbiamo a disposizione solo risorse limitate (che essenzialmente sono tempo e memoria). Questo fatto non ci permette di processare con facilità espressioni arbitrariamente incassate come (29).c:

(29) a. il topo è stato comprato da mio padre
b. il topo che il gatto si mangiò è stato comprato da mio padre
c. ??il topo che il gatto che il cane morse si mangiò è stato comprato da mio padre

Eppure la regola di "espansione" che ci premette di passare da (29).a a (29).b, cioè l'introduzione di una frase relativa che modifica il soggetto della frase, è un'operazione completamente legittima e infatti

produce (29).b che è un'espressione del tutto accettabile. La stessa operazione però produce risultati abbastanza inaccettabili passando da (29).b a (29).c. Chomsky spiega questo fenomeno ricorrendo appunto alla distinzione tra *competence* (*competenza*) e *performance* (*esecuzione*): la *competence* esprime la nostra conoscenza linguistica ideale, cioè quell'insieme di regole/principi che ci permettono di combinare le primitive linguistiche ai vari livelli, mentre la *performance* è la competence in uso, cioè il reale utilizzo che le persone fanno della loro conoscenza linguistica utilizzando le (limitate) risorse di memoria e di tempo di cui dispongono. Se non avessimo limiti di memoria, secondo questo punto di vista, (29).c non ci sembrerebbe strana (attenzione: "strana" anche in questo contesto non corrisponde ad agrammaticale; qua l'intuizione è un po' meno evidente visto che saremmo portati a pensare che mai un parlante nativo potrebbe produrre o comprendere un'espressione del genere. Tale frase è però generata applicando una regola che abbiamo considerato completamente grammaticale, quindi il risultato dell'applicazione di tale regola deve produrre risultati grammaticali per definizione. Se questo risultato non ci convince dovremmo rimuovere la regola che ci permette di modificare il soggetto con frasi relative, ma questo escluderebbe anche (29).b dalla nostra lingua, che, come possiamo facilmente constatare, è invece un'espressione piuttosto accettabile).

[Processing] Oltre al concetto di perfomance, che ci permette quindi di escludere certe espressioni linguistiche, come (29).c, in linea di principio grammaticali, ma che risultano comunque difficili da processare, si affianca quello di *processing*: per *processing* si intende l'uso effettivo della competence per comprendere e produrre una frase, cioè quella precise serie di regole/proprietà e priorità di combinazione che ci permettono, ad esempio, di spiegare perché certe espressioni sono più difficili da processare rispetto ad altre ugualmente grammaticali ((29).b Vs. (29).c) (quindi di render conto delle difficoltà di performance) e perché, in caso di ambiguità, certe interpretazioni risultano più accessibili di altre.

[Parsing] Un tipico problema di processing è il *parsing*, cioè quella serie di strategie che ci permettono di assegnare una struttura (*parsing* deriva dal latino *pars*, cioè parte, richiamando quindi il processo di suddivisione di un'espressione in parti o costituenti) ad una sequenza di suoni/parole accettabile per la nostra lingua. Quindi fare parsing corrisponde ad accettare o rifiutare una stringa di testo (sequenza di parole) e, nel caso di accettazione, assegnare a questa stringa almeno una struttura che ci permetta di interpretarla. Vale la pena di sottolineare "almeno" poiché nella possibilità di assegnare ad una stessa stringa di testo più strutture risiede tutta la difficoltà del task.

[Livelli di ambiguità] La complessità del problema deriva in effetti dal fatto che la mappatura tra "parole" ed "etichette" non è sempre univoca; prendiamo una famosa frase ambigua:

(30) la vecchia legge la regola

Le due interpretazioni possibili sono le seguenti:

(31) a. una legge antica controlla qualcosa (ad esempio un determinato contesto)
b. una vecchia signora sta leggendo una certa direttiva

Assegnare a questa frase una struttura richiede prima di tutto l'assegnazione ad ogni token di un Part of Speech (o PoS, etichetta pre-terminale, vedere 1.1). Assumiamo che questa sia la nostra lista di PoS:

(32) D (articolo), Pro (pronome), A (aggettivo), N (nome), V (verbo).

Si può facilmente notare che tutte le parole della frase in (30) sono ambigue tra due PoS:

(33) la = D/Pro
vecchia = N/A
legge = N/V
regola = N/V

Si parla in questo caso di *ambiguità lessicale*, cioè una stessa parola (o token) è ambigua tra due categorie morfologiche distinte (o *PoS*).

L'ambiguità lessicale si ripercuote chiaramente anche sulla struttura della frase, ad esempio scegliere di interpretare "vecchia" come nome o come aggettivo determina una suddivisione in costituenti ben precisa a livello sintattico.

Ci sono casi, come il seguente, in cui non basta rimuovere l'ambiguità lessicale per avere accesso a soluzioni strutturali distinte:

(34) Gianni ha visto Maria con il cannocchiale

Come accennato in 4.1, la frase potrebbe significare sia che Gianni ha usato un cannocchiale per vedere Maria, (35).a, sia che Maria aveva un cannocchiale quando Gianni l'ha vista (35).b:

(35) a. [Gianni ha visto [Maria] [con il cannocchiale]]
b. [Gianni ha visto [Maria [con il cannocchiale]]]

La visualizzazione della struttura, come mostrato in (35), permette di esprimere esattamente questa *ambiguità* che si definisce *sintattica*, poiché legata alla diversa assegnazione dei rapporti di inclusione/modificazione tra i costituenti.

Esattamente questo tipo di ambiguità è stato estensivamente studiato sia da un punto di vista psicolinguistico (Frazier e Fodor 1978), che da un punto di vista computazionale (Brill e Resnik 1994) ed è noto come il problema di attaccamento dei sintagmi preposizionali (*PP*, Prepositional Phrase, *attachment*).

Per avere un'idea della "gravità" del problema basta aggiungere un altro sintagma preposizionale:

(36) Gianni ha visto Maria nel parco con il cannocchiale

In questo caso ci sono almeno cinque possibili interpretazioni ed è facile capire come aggiungendo ulteriori sintagmi preposizionali le cose peggiorino in modo (quasi) esponenziale:

(37) a. [G. ha visto [M.] [nel parco [con il cannocchiale]]]
G. era nel parco dove c'era un cannocchiale e guardava M. (che non era nel parco)
b. [G. ha visto [M. [nel parco [con il cannocchiale]]]]
G. guardava M. che era nel parco dove c'era un cannocchiale
c. [G. ha visto [M. [nel parco][con il cannocchiale]]]]
G. guardava M. che aveva un cannocchiale ed era nel parco
d. [G. ha visto [M.] [nel parco][con il cannocchiale]]
G. usava un cannocchiale mentre era nel parco per guardare M. (che non era nel parco)
e. [G. ha visto [M. [nel parco]][con il cannocchiale]]
G. usava un cannocchiale per guardare M. che era nel parco

C'è infine una terza tipologia di ambiguità di cui abbiamo già brevemente parlato nell'introduzione di questo capitolo: l'*ambiguità semantica*. La parola "pianta", può significare sia "parte del piede" che "vegetale". La categoria grammaticale (PoS) è la stessa nei due casi (N), quindi non siamo di fronte ad un'ambiguità lessicale; d'altra parte nelle espressioni "mostrami [la pianta [del piede]]" e "annaffia [la pianta]", il sintagma "la pianta" è in entrambi i casi un sintagma nominale (NP), quindi non si può parlare neppure di ambiguità sintattica. Ci dobbiamo quindi convincere del fatto che questo terzo livello è ragionevolmente distinto e indipendente dagli altri due.

In conclusione: un problema è più difficile se contemporaneamente devo valutare più possibilità, tutte ugualmente plausibili. Scelte multiple tra cui non ho euristiche di scelta portano al non-determinismo e ci costringono a valutare più alternative. L'ambiguità è uno dei fattori che conduce al non-determinismo ed è presente ad ogni livello della descrizione linguistica. Nella prossima sezione ci concentreremo su questi livelli di descrizione.

4.2. Come si descrive una lingua

Le varie proprietà linguistiche che abbiamo discusso nella sezione precedente ci servono per descrivere esplicitamente quella che è la nostra lingua naturale: in questa sezione vedremo come le categorie individuate ci permetteranno di annotare un testo in modo da trasformarlo da semplice insieme di parole in fonte esplicita di informazione linguistica (morfosintattica e semantica). Questo passo è fondamentale per creare lessici e grammatiche che poi saranno le conoscenze che la macchina potrà utilizzare per disambiguare testi, per tentare traduzioni automatiche da una lingua all'altra e per recuperare informazioni specifiche.

Da un punto di vista teorico, possedere una grammatica (ad esempio una grammatica a struttura sintagmatica come quella che abbiamo descritto nella sezione precedente) ci permette di circoscrivere esattamente l'insieme di espressioni ben formate che fanno parte del nostro linguaggio e di capire come si articola il significato di tali espressioni riuscendo ad interpretarlo anche per frasi mai viste precedentemente. **[Livelli di adeguatezza]** Se abbiamo fatto un buon lavoro, cioè se la nostra grammatica effettivamente descrive una realtà linguistica interessante, diremo che la nostra grammatica è *adeguata*. Tale adeguatezza potrà essere a tre livelli: parleremo di *adeguatezza osservativa* se la lingua definita dalla grammatica coincide esattamente con quella che si intende descrivere, cioè se la nostra grammatica genera esattamente tutte e sole le stringhe che sono frasi dell'Italiano, senza assegnare loro nessuna particolare struttura; si tratterà di *adeguatezza descrittiva* se anche l'analisi grammaticale proposta è in linea con le intuizioni linguistiche dei parlanti fornendo descrizioni strutturali (ad esempio alberi sintattici) adeguate delle frasi accettabili; infine parleremo di *adeguatezza esplicativa* se i dispositivi generativi utilizzati soddisfano criteri di plausibilità psicolinguistica e riproducono realmente i meccanismi operanti nell'attività linguistica del parlante. Una grammatica si dice esplicativa quando rende conto anche dell'apprendibilità della lingua, ovvero di come un bambino esposto ad un determinato idioma riesca effettivamente ad apprenderlo. Questo terzo livello di adeguatezza include logicamente gli altri due ma è molto difficile da raggiungere: una teoria grammaticale esplicativamente adeguata è una teoria necessariamente molto complessa che considera un vasto numero di fenomeni (quali appunto i dati sull'acquisizione del linguaggio nei bambini) che non abbiamo modo di discutere in queste pagine. L'obiettivo dei modelli grammaticali discussi qui sarà quindi quello di raggiungere il livello descrittivo poiché, come vedremo nelle sezioni successive, una corretta descrizione strutturale sarà fondamentale per la comprensione di espressioni ambigue.

Cominciamo adesso ad analizzare i livelli descrittivi della nostra grammatica partendo dal lessico, per giungere ad una descrizione utile delle proprietà linguistiche e ad una loro misurazione quantitativa ottenibile analizzando raccolte di testi opportunamente annotati.

4.2.1. Il lessico e la rappresentazione del significato

Nel paragrafo 4.1.2 avevamo definito la competenza linguistica come un'insieme di conoscenze definibili a diversi livelli descrivibili con un grado di ragionevole indipendenza. **[Descrizione fonetica]** Prendiamo ad esempio il livello fonetico: i suoni significativi in ogni lingua sono molti, ma di certo non sono infiniti; studi psicolinguistici hanno mostrato come la percezione del continuum acustico tra un fonema sordo (la "p" ad esempio nella sillaba "pa") ed uno sonoro (la "d" nella sillaba "da") sia strettamente categoriale (Phillips et al. 2000); tale risultato si basa sul fatto che registrando l'attività neuronale (attraverso tecniche di magneto-encefalografia, MEG, che permettono di visualizzare precisi pattern di attivazione, *Magnetic Mismatch Field, MMF*, quando due stimoli sono percepiti in modo distinto) di persone esposte a sillabe all'interno delle quali si variava artificialmente l'intervallo tra la consonante occlusiva e la sonorizzazione della vocale a

seguire (*Voice Onset Time, VOT*) in modo da ottenere moltissime varianti all'interno di un intervallo tra il fonema sonoro e quello sordo, il pattern di attivazione corticale mostrava chiaramente come sopra un certo intervallo l'attivazione era sempre uniforme (cioè veniva comunque percepito un fonema sonoro anche se i vari suoni differivano sostanzialmente lungo un continuum acustico), sistematicamente diverso da quello che veniva registrato sotto tale intervallo (nonostante le variazioni, sotto questa soglia veniva percepito sistematicamente un fonema sordo). **[Trascrizione fonetica e l'International Phonetic Alphabet, IPA]** Pur essendo finiti, i fonemi producibili in una lingua naturale qualsiasi sono comunque molti ed i caratteri utilizzati nell'alfabeto latino standard non sono sufficienti ad esprimerli univocamente. Per ovviare a questo problema di ambiguità è stato ideato uno standard chiamato *Alfabeto Fonetico Internazionale* (*International Phonetic Alphabet*, *IPA*) che enumera un centinaio di suoni (*foni*) principali (circa una settantina di consonanti e una trentina di vocali) che descrivono tutti i possibili fonemi presenti in tutte le lingue naturali. L'organizzazione di questi suoni avviene per opposizioni che ne identificano i *tratti distintivi* e solitamente descrivono sia proprietà acustiche (ad esempio la distinzione tra sorda "pa" Vs. sonora "ba") sia i modi che i luoghi dell'articolazione (consonanti occlusive, nasali, articolate usando le labbra, la lingua sui denti, sugli alveoli, sul palato ecc.). Ogni lingua seleziona da questo insieme un set limitato di fonemi; ad esempio per descrivere adeguatamente l'Italiano abbiamo bisogno di non più di 23 consonanti e di 7 vocali:

		bilabiale	**labiodentale**	**dentale**	**alveolare**	**alveopalatale**	**palatale**	**velare**
occlusive	sorda	p		t				k
	sonora	b		d				g
fricative	sorda		f	s		ʃ		
	sonora		v	z				
affricate	sorda			ts		tʃ		
	sonora			dz		dʒ		
nasali		m			n		ŋ	
liquide					r - l		ʎ	
semi-consonanti							j	w (labiovelare)

Tabella 1. Le consonanti dell'Italiano

	anteriore	**centrale**	**posteriore**
alta	i		u
	e		o
	ɛ	ɔ	
bassa		a	

Tabella 2. Le vocali dell'Italiano

Poiché i 30 fonemi principali dell'italiano vengono mappati in 21 lettere dell'alfabeto, più fonemi saranno trascritti ambiguamente con lo stesso carattere: è questo il caso della "s" di "rosa" (/rɔ*z*a/) e la "s" di "strada" (/*s*trada/) oppure la "e" di pesca (/p*e*ska/, attività sportiva) e quella di "pesca" (/p*ɛ*ska/, frutto).

Ecco alcuni esempi di trascrizione in fonemi per familiarizzare con la mappatura tra stringhe e suoni in Italiano e meglio comprendere le tabelle 1 e 2 (il diacritico ":" indica il raddoppiamento del fonema che precede): gola /gola/ , gelo /dʒɛlo/ , cara /kara/ , cera /tʃera/ , notte /nɔt:e/ , inglese /inglese/ , aglio /aʎ:o/ , scena /ʃɛna/ , gnoseologico /gnoseolɔgiko/ , gnomo /ŋɔmo/ , uomo /wɔmo/, imbuto /imbuto/, piano /pjano/.

Queste idiosincrasie dovrebbero convincerci del fatto che il livello fonetico è un livello di descrizione aggiuntiva al livello ortografico che potremmo voler includere nel nostro lessico. Questa scelta sarà assolutamente necessaria se volessimo istruire il nostro computer a pronunciare correttamente certe parole; la disciplina, interna alla Linguistica Computazionale, che studia e cerca di mappare in modo computazionalmente efficiente fonemi e grafemi, oltre alla loro segmentazione in parole e frasi, è nota come *Speech Recognition* (*SR*) per la parte di riconoscimento del parlato, appunto, e come *Speech Synthesis* o *Text-to-Speech*, *TTS*) per la parte di generazione.

[Descrizione morfo-sintattica] Sempre in 4.1.2 abbiamo parlato del livello *morfemico* e del livello delle *parole*: abbiamo definito i *morfemi* come i più piccoli raggruppamenti di fonemi dotati di significato, quindi entità portatrici di informazioni grammaticali come ad esempio i tratti di *accordo di genere e numero* ("un-*a* pall-*a* ross-*a*"), l'espressione di *tempo* ("mang-*er*-ai") e *modo* verbale (mang-*erest*-i) ecc. oltre ovviamente alle radici verbali ("*corr*-ere") e nominali ("*pall*-a") rispettivamente associabili a certi tratti categoriali (*verbo* e *nome*) e sottocategoriali (*verbo intransitivo*, *nome comune di cosa*). In lingue come l'Italiano non è sempre possibile associare ai singoli morfemi dei tratti univoci: ad esempio nella flessione dell'imperfetto in italiano "mang-ia-*v*-a" viene contemporaneamente espresso sia un tratto temporale, *passato* (l'evento è avvenuto nel passato rispetto al momento del'enunciazione), che uno aspettuale, *imperfettivo* (l'evento predicato è "in corso", incompleto al momento dell'enunciazione); oltre a questo, i singoli morfemi non sono univocamente associabili a tratti morfosintattici e semantici: la "-*i*" di "corr-*i*" esprime la seconda persona singolare, tempo presente, mentre la stessa "-*i*" in "ram-*i*" esprime il plurale maschile. Registrare quindi a livello di lessico tutti i possibili morfemi è un compito abbastanza complesso e il risultato potrebbe essere un dizionario notevolmente ambiguo se non aggiungiamo regole precise di combinazione e quindi di disambiguazione. **[L'ipotesi della ridondanza minima]** D'altra parte, sapendo come associare i morfemi secondo regole che chiamiamo di *morfologia flessiva* (io mangi-*o*, tu mang-*i* egli mangi-*a*) o *derivazionale* (fatic-*a*, fatic-*are*, fatic-*oso*, fatic-*osa-mente*) riusciamo ad avere un lessico che non presenta ridondanza e che esprime in modo compatto e potenzialmente efficiente la nostra competenza lessicale.

Come si può osservare in tabella 3 per ottenere venti parole italiane, potrebbero bastare quattro radici e sette flessioni verbali, più qualche regola di combinazione che permetta di gestire anche le eventuali eccezioni ("corso" e non "corr-ato" o "corr-uto").

	mangi -	sogn -	corr -	googl -
-are/ere	mangi-are	sogn-are	corr-ere	googl-are
-o	mangi-o	sogn-o	corr-o	googl-o
-i	mangi-i	sogn-i	corr-i	googl-i
-a/-e	mangi-a	sogn-a	corr-e	googl-a
-ato	mangi-ato	sogn-ato	*corr-ato (corso)	googl-ato

Tabella 3. Lessico "generativo" scomposto in morfemi

E' importante sottolineare che ogni volta che introduciamo una nuova radice verbale nella tabella 3, otteniamo ben cinque nuove parole dell'italiano; inoltre per ogni nuova radice verbale introdotta nel nostro lessico (ad esempio neologismi come "*googl*-"), il computer saprà esattamente come fletterla anche senza essere mai stato esposto ad una versione flessa prima, esattamente come succede per i parlanti nativi ("prova a *googl*-ar-lo!"). In questo senso possiamo parlare di lessico "generativo". Un lessico strutturato in radici e flessioni risponde quindi ad una serie di esigenze interessanti quali la compattezza della

rappresentazione dell'informazione linguistica (definita anche come *minumum redundancy*, Jurafksy e Martin 2008) e la sua plausibilità psicolinguistica (ad esempio, in inglese, sono stati registrati errori di pronuncia del tipo “eas-*y* enough-*ly*” in cui la flessione corretta “easi-*ly* enough” viene invertita, Fromkin e Ratner 1998; questo sembra suggerire che il lessico mentale debba contenere alcune informazioni sulla struttura morfologica delle parole rappresentate). Suddividere il lessico in morfemi permette inoltre di trattare in modo più semplice le lingue agglutinanti, cioè quelle lingue in cui i morfemi possono comporsi in sequenze lunghissime formando parole estremamente complesse (in Turco otterremmo circa $600x10^6$ entrate lessicali se volessimo considerare ogni plausibile combinazione morfemica).
Lessici che cercano di rappresentare la struttura morfemica delle parole sono in realtà macchine a stati finiti (*FSA*, vedi cap. 1) o trasduttori (*FST*, dispositivi che essenzialmente corrispondono a due FSA che procedono in parallelo, il primo che riconosce una stringa di testo, il secondo che la riscrive, Koskenniemi 1984): dispositivi che permettono di definire pattern di identificazione (vedi anche 4.3.4.1) che circoscrivono un insieme di stringhe che appunto corrispondono a radici e flessioni; combinando opportunamente i vari riconoscitori di radici e flessioni si ottiene un lessico. Un esempio di analizzatore morfologico che sfrutta questo approccio è descritto in figura 3: seguendo una serie di convenzioni grafiche, gli stati di un automa sono visualizzati come cerchi; q_0 è lo stato iniziale, q_f quello finale (segnalato dal contorno doppio), sugli archi sono indicati degli insiemi di caratteri (cioè liste di sequenze di caratteri) che permettono di avanzare da uno stato all’altro, nel senso della freccia, fino allo stato finale. Se tale stato finale viene raggiunto, il riconoscimento (o la generazione) di una parola del nostro dizionario è avvenuta. Ad esempio, “radici nominali maschili” sarà un insieme di stringhe come {“tavol”, “cartell”, “ors”, ...} mentre “flassioni nominali maschili” sarà un insieme composto da {“o”, “i”}. Tale automa è ovviamente molto semplificato ma l’aggiunta di ulteriori stati (a cui si arriva, ad esempio selezionando delle “radici nominali femminili” e da lì a q_f passando per le “flessioni nominali femminili”) non aggiunge niente di teoricamente nuovo al meccanismo fin qui descritto.

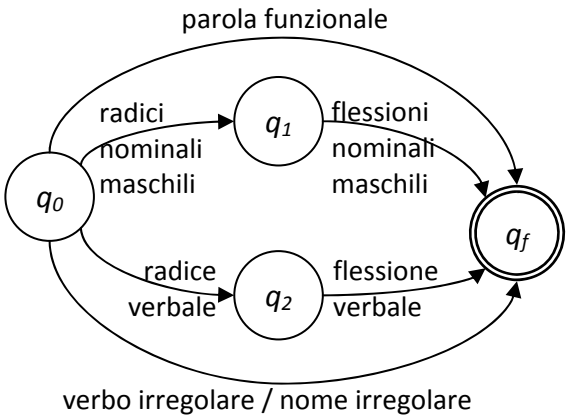


Figura 3. Esempio semplificato di automa a stati finiti per il riconoscimento di parole e morfemi nominali e verbali.

Alla fine, il nostro dizionario sarà composto da classi di sottostringhe (radici nominali, verbali e aggettivali, flessioni corrisponenti, liste di eccezioni, clitici, ecc.) concatenabili secondo le regole del nostro automa a stati finiti.

[L'ipotesi della lista esaustiva] Un altro approccio computazionale al lessico è quello che segue l'*ipotesi della lista esaustiva* (*full listing hypothesis*, Jurafsky e Martin 2008); seguendo questa idea ogni occorrenza possibile di una parola della nostra lingua, viene registrata singolarmente all'interno del lessico e ad essa vengono associate tutte le informazioni morfo-sintattiche ritenute rilevanti, quali il lemma di provenienza, i tratti di accordo e di tempo ecc.:

(38) mangiavo (*lemma*:mangiare, *tempo*:imperfetto, *modo*: indicativo, *persona*:1, numero:sing. …)
mangiavi (*lemma*:mangiare, *tempo*:imperfetto, *modo*: indicativo, *persona*:2, numero:sing. …)
mangiava (*lemma*:mangiare, *tempo*:imperfetto, *modo*: indicativo, *persona*:3, numero:sing. ...

Un esempio di lessico così strutturato per l'italiano è *Morph-it!* (Zanchetta e Baroni 2005): tale risorsa include oltre 500.000 entrate lessicali che corrispondono a circa 35.000 lemmi distinti ed è ottenuta annotando semiautomaticamente un corpus (il corpus "Repubblica" contenente la raccolta completa di articoli di giornale comparsa sul quotidiano Repubblica dal 1985 al 2000, per un totale di circa 380 milioni di parole, Baroni et al. 2004). Il lessico così creato si presenta o compilato in forma di automa a stati finiti (rif. cap. 1) oppure in forma di file testuale formattato con tabulazione che delimita i vari campi (*Tab Separated Values*, *TSV*); in quest'ultimo formato le singole entrate hanno la seguente struttura:

(39)

token	*lemma*	*categoria:tratti*
rimpinzeremmo	rimpinzare	VER:cond+pre+1+p
abominevoli	abominevole	ADJ:pos+m+p
dabbenaggine	dabbenaggine	NOUN-F:s
ostensibilmente	ostensibilmente	ADV

Lo spazio occupato dal file non compresso è di circa 19 MB e quindi richiede un certo sforzo computazionale per essere utilizzato ed editato. I vantaggi di una simile risorsa sono comunque la codifica di alto livello e quindi la relativa facilità di aggiornamento e accesso anche da parte di utenti senza particolari competenze linguistiche e/o informatiche.

4.2.1.1 L'annotazione lessicale e le categorie morfo-sintattiche

[L'inventario dei tratti morfo-sintattici] Contrariamente al livello *fonetico*, a livello *morfemico* non è semplice definire un inventario finito ed universale che enumeri tutti i possibili tratti ed i relativi valori esprimibili dai morfemi in ogni lingua. Questo in parte è dovuto alle diverse teorie linguistiche che cercano di segmentare in modo diverso le categorie morfo-sintattiche (PoS) in modo da cogliere opposizioni sempre più sottili e significative ("rossa" e "francese" devono essere classificati semplicemente come "aggettivi" o il primo va classificato come aggettivo che esprime un colore e il secondo come un aggettivo che esprime una nazionalità?), in parte perché avendo un inventario di categorie/tratti ridotto, le risorse necessarie per analizzare un testo sono minori ed assegnare una determinata categoria ad un determinato token è più semplice.

Vari standard sono stati proposti con il fine di armonizzare e valutare le varie risorse linguistiche sviluppate. Una serie di importanti linee guida per la definizione di questo insieme di tratti (che in gergo tecnico si definisce *tagset*) è proposto da Monachini e Calzolari (1996) seguendo le direttive del progetto *EAGLES* (Expert Advisory Group on Language Engineering Standards), un progetto di iniziativa della Commissione Europea con l'obiettivo di velocizzare e uniformare lo sviluppo di risorse linguistiche quali corpora e lessici computazionali, definendo specifiche generali, standard di valutazione e formalismi grammaticali in modo da rendere maggiormente riutilizzabili ed aggiornabili tali risorse. Tali linee guida sono state recepite in vari standard per l'italiano; tra questi *TANL* (Text Analytics and Natural Language), standard utilizzato di recente per valutare le performance dei sistemi di *PoS Tagging* (Attardi e Simi 2009). Lo standard prevede 14 macrocategorie sintattiche (tabella 4) e 36 categorie morfosintattiche più sottili (interamente riportate in Tabella 5, le tabelle sono un riadattamento di quelle in Attardi e Simi 2009).

Tag	Descrizione	Esempio
A	aggettivo	bello
B	avverbio	velocemente
C	congiunzione	e, o
D	determinante	questo, quello
E	preposizione	di, a, da
F	punteggiatura	. , ! ?
I	Interiezione	beh
N	numerale	uno, due
P	pronome	suo, io
R	articolo	il, lo
S	nome	cane
T	predeterminante	tutti, ogni
V	verbo	corre
X	classe residuale	SpA

Tabella 4. Macrocategorie morfo-sintattiche

categoria	descrizione	esempi	contesto d'uso
A	aggettivo	bello, buono, bravo	una **bella** passeggiata, una persona **brava**
AP	aggettivo possessivo	mio, tuo, nostro	a **mio** parere, il **tuo** libro
B	avverbio	bene, fortemente, malissimo, domani	arrivo **domani** sto **bene**
BN	avverbio negativo	non	**non** sto bene
CC	congiunzione coordinativa	e, o, ma	i libri **e** i quaderni, vengo **ma** non rimango
CS	congiunzione subordinativa	mentre, quando	**quando** ho finito vengo, **mentre** parlava rideva
DD	determinante dimostrativo	questo, codesto, quello	**questo** denaro, **quella** famiglia
DE	determinante esclamativo	che, quale, quanto	**che** disastro! **quale** catastrofe!
DI	determinante indefinito	alcuno, certo, tale, parecchio, qualsiasi	**alcune** telefonate, **parecchi** giornali, **qualsiasi** persona
DQ	determinante interrogativo	cui, quale	i **cui** libri, **quale** intervista
DR	determinante relativo	che, quale, quanto	**Che** cosa, **quanta** strada, **quale** formazione
E	preposizione	di, a, da, in, su, attraverso, verso, prima_di	**a** casa, **prima_di** giorno **verso** sera
EA	preposizione articolata	alla, del, nei	**nel** posto
FB	punteggiatura parentetica	( ) [ ] { } -_ ` ' ‘ ’ " “ ” « »	**(**sempre**)**
FC	punteggiatura di fine frase	. -: ;	segue**:**
FF	virgola, trattino	,	carta**,** penna, 30**-**40 persone
FS	punteggiatura di fine espressione	. ? ! ...	cosa**?**
I	interiezione	ahimè, beh, ecco, grazie	**Beh**, che vuoi?
N	numero cardinale	uno, due, cento, mille, 28	**due** partite, **28** anni
NO	numero ordinale	primo, secondo, centesimo	**secondo** posto
PC	pronome clitico	mi, ti, ci, si, te, ne, lo, la, gli	**me ne** vado, **si** sono rotti, **gli** parlo
PD	pronome dimostrativo	questo, quello, costui, ciò	**quello** di Roma, **costui** è innocente

PE	pronome personale	io, tu, egli, noi, voi	**io** parto, **noi** scriviamo
PI	pronome indefinito	chiunque, ognuno, molto	**chiunque** venga, i diritti di **ognuno**
PP	pronome possessivo	mio, tuo, suo, loro, proprio	il **mio** è qui, più bella della **loro**
PQ	pronome interrogativo	che, chi, quanto	non so **chi** parta **quanto** costa? **che** ha fatto ieri?
PR	pronome relativo	che, cui, quale	ciò **che** dice, il **quale** afferma, a **cui** parlo
RD	articolo determinativo	il, lo, la, i, gli, le	**il** libro, **i** gatti
RI	articolo indeterminativo	uno, un, una	**un** amico **una** bambina
S	nome comune	amico, insegnante, verità	l'**amico**, la **verità**
SA	abbreviazione	ndr, a.C., d.o.c., km	30 **km**, sesto secolo **a.C.**
SP	nome proprio	Monica, Pisa, Fiat, Sardegna	**Monica** scrive
T	predeterminante	tutto, entrambi	**tutto** il giorno, **entrambi** i bambini
V	verbo principale	mangio, passato, camminando	**mangio** la sera, il peggio è **passato,** ho **scritto** una lettera
VA	verbo ausiliario	avere, essere, venire	il peggio **è** passato, **ho** scritto una lettera, **viene** fatto domani
VM	verbo modale	volere, potere, dovere, solere	non **posso** venire, **vuole** comprare il libro
X	classe residua	formule, parole non classificate, simboli alfabetici, ecc.	distanziare di **43''**

Tabella 5. Categorie morfo-sintattiche più dettagliate

A queste categorie vengono poi associati i seguenti tratti morfologici:

tratti	**valore**
genere	m (maschile), f (femminile), n (non specificato)
numero	s (singolare), p (plurale), n (non specificato)
persona	1 (primo), 2 (seconda), 3 (terza)
modo	i (indicativo), m (imperativo), c (congiuntivo), d (condizionale), g (gerundio), f (infinito), p (participio)
tempo	p (presente), i (imperfetto), s (passato), f (futuro)
clitico	c segnala la presenza di un clitico incorporato.

Tabella 6. Tratti e valori morfo-sintattici

Combinando le classi morfosintattiche con i tratti morfologici si ottengono 328 etichette (*tags*). Ecco ad esempio i tags aggettivali utilizzati nell'etichettatura:

Categoria principale	**Categoria con tratti morfologici**	**Esempio**
A (aggettivo)	Ams (agg. masc. sing.)	tossico, doppio, italiano …
	Amp (agg. masc. plur.)	chimici, tossici, giudiziari ...
	Afs (agg. femm. sing.)	moderna, splendida, clamorosa ...
	Afp (agg. masc. plur.)	numerose, belle, antiche ...
	Ans (agg. genere non spec. sing.)	speciale, londinese, lunghista ...
	Anp (agg. genere non spec. plur.)	trasparenti, mondiali, pesanti, naturali ...
	Ann (agg. genere e numero non spec.)	top_secret, ex, pari ...

Tabella 7. Esempio di combinazione di una categoria (A) con i tratti/valori morfo-sintattici

Un testo annotato con queste etichette si presenta di solito su due colonne separate da tabulazione, nella prima colonna si trova il token, cioè la parola trovata nel testo, nella seconda colonna la categoria morfo-sintattica di riferimento (*PoS tag*):

token	PoS Tag (TANL tagset)
A	E
ben	B
pensarci	Vfc
,	FF
l'	RDns
intervista	Sfs
dell'	EAns
on.	SA
Formica	SP
è	VAip3s
stata	VApsfs
accolta	Vpsfs
in	E
genere	Sms
con	E
disinteresse	Sms
.	FS

Tabella 8. Esempio di testo annotato per categorie morfo-sintattiche (*PoS tagging*)

4.2.1.2 La struttura globale del lessico

Come abbiamo visto, un lessico computazionale può essere concepito per rispondere ad esigenze diverse: potremmo ad esempio avere modelli lessicali molto semplici e facilmente accessibili/aggiornabili ma computazionalmente non molto euristici e/o efficienti (full listing hypothesis) oppure lessici psicolinguisticamente più plausibili, computazionalmente più complessi e difficili da aggiornare, ma al tempo stesso più efficienti in termini di risorse di memoria e di calcolo richieste per venire processati (minimum redundancy hypothesis). Fin qui però ci siamo occupati solo della struttura informativa delle singole entrate lessicali; in effetti anche la *struttura globale* del nostro lessico merita una certa attenzione: l'ipotesi più semplice che possiamo fare è che nel nostro lessico le parole siano ordinate in ordine alfabetico. Questo le rende facilmente individuabili e ogni nuova parola potrà essere aggiunta in una precisa posizione del nostro lessico usando un semplice algoritmo. In linea di principio esistono comunque altre strategie per ordinare le nostre entrate lessicali e per gestire eventuali relazioni tra parole: ad esempio, se disponiamo di una collezione di testi, questi testi potranno essere ispezionati e per ogni parola presente nei vari documenti potremmo contare quante volte occorre globalmente nella collezione. Prendiamo un piccolo testo come rappresentativo della nostra "collezione" di parole:

(40) Agilulfo aveva sempre bisogno d'applicarsi a un esercizio d'esattezza: contare oggetti, ordinarli in figure geometriche, risolvere problemi d'aritmetica. Se invece il mondo intorno sfumava nell'incerto, nell'ambiguo, anch'egli si sentiva annegare in questa morbida penombra, non riusciva più a far affiorare dal vuoto un pensiero distinto, uno scatto di decisione, un puntiglio. Stava male: erano quelli i momenti in cui si sentiva venir meno; alle volte solo a costo di uno sforzo estremo riusciva a non dissolversi. Allora si metteva a contare: foglie, pietre, lance, pigne, qualsiasi cosa avesse davanti. O a metterle in fila, a ordinarle in quadrati o piramidi.

> L'applicarsi a queste esatte occupazioni gli permetteva di vincere il malessere, d'assorbire la scontentezza, l'inquietudine e il marasma, e di riprendere la lucidità e compostezza abituali
> (Italo Calvino, Il cavaliere inesistente, 1959)

[Tokens, Types e frequenze] Questo testo è composto da 135 occorrenze di parole (*tokens*) corrispondenti a 100 forme distinte (*types*); ad esempio il token "a" compare 8 volte, "in" 5 volte e così via, come riportato in tabella per le prime 20 forme.

forma	occorrenze
a	8
in	5
d'	4
di	4
il	3
un	3
e	3
si	3
sentiva	2
l'	2
contare	2
applicarsi	2
nell'	2
o	2
non	2
la	2
riusciva	2
uno	2
erano	1
venir	1

Tabella 9. Tabella delle frequenze delle varie parole/forme all'interno di un testo

Questi indici di frequenza possono esprimere un ulteriore parametro per organizzare/ordinare il nostro lessico. Torneremo sull'importanza di queste informazioni statistiche nel paragrafo 4.2.3. **[Relazioni semantiche]** Adesso concentriamoci su un ulteriore modo per organizzare globalmente le nostre entrate lessicali: le parole "casa", "edificio" e "abitazione" sono in qualche modo intuitivo più *collegate* di quanto non lo siano "casa", "cane" e "cavo" e questa "vicinanza" è chiaramente non evidente osservando l'ordine alfabetico o quello di frequenza. In effetti tra "casa" ed "edificio" sussiste una relazione semantica chiamata *iponimia*: "casa" è un termine più specifico di "edificio", cioè il termine "casa" è applicabile solo ad un sottoinsieme di oggetti per cui si può propriamente dire che sono "edifici" (la relazione inversa si definisce di *iperonimia*: "fiore" è un *iperonimo* di "orchidea"). Tra "casa" ed "abitazione" sussiste invece una relazione di *sinonimia* (tale relazione è ovviamente bidirezionale, esiste comunque anche una relazione di "polarità inversa" definite di *antinomia* che lega ad esempio l'aggettivo "vuoto" e "pieno"). Altre relazioni interessanti possono essere quella di *parte-di* (*meronimia*) che lega ad esempio la parola "braccio" a "corpo" o di diversa modalità di un'azione/evento (*troponimia*): "correre" è un *troponimo* di "camminare". Questa riflessione ci porta a considerare una rete che connette in vario modo gli elementi del nostro lessico e che risulta invisibile partendo dalla forma ortografica, dagli indici di frequenza e dalle informazioni morfo-sintattiche associate alle parole. Sfruttando queste relazioni si può definire una struttura globale del lessico

adatta a rispondere in modo psicolinguisticamente plausibile alla domanda: quale è il significato di una parola? La risposta è data dalla serie di link che legano il significato associato ad una parola ad altri significati di altre parole. **[WordNet]** La risorsa sviluppata dal gruppo di Fellbaum e Miller (Princeton University) parte da questa idea e si chiama *WordNet* (Fellbaum 1998). Attualmente, giunta alla sua versione 3.0, si presenta come una struttura dati di circa 155.000 parole che esprimono quasi 118.000 significati distinti (*synset*) organizzati in circa 207.000 relazioni semantiche. WordNet è spesso considerato un'importante base di conoscenza (*Lexical Knowledge Base*, *LBK*) e ha avuto un notevole successo presso la comunità scientifica che ha cercato di riprodurlo in diverse lingue e di allinearlo alla versione inglese. Da segnalare, in questo senso, è il progetto *EuroWordNet*: progetto finanziato dall'Unione Europea che ha coinvolto 12 partner europei e ha permesso di riprodurre una rete completa di lemmi sul modello inglese per Ceco, Estone, Fiammingo, Francese, Italiano, Spagnolo e Tedesco. Si è quindi tentato di allineare ogni distinta struttura con un'interlingua semantica basata sul WordNet inglese (*Inter-Lingual Index*). In modo simile, il progetto MultiWordNet dell'ITC-IRST (Istituto per la Ricerca Scientifica e Tecnologica dell' Istituto Trentino di Cultura, adesso Fondazione Bruno Kessler, FBK) ha permesso di sviluppare una risorsa multilingue basata su questa rete semantica (Italiano, Spagnolo, Portoghese, Ebraico, Rumeno e Latino). In MultiWordNet si è dato priorità alla struttura inglese e su questa si è iniziato il processo di traduzione e individuazione dei gap cross-linguistici in ciascuna lingua (ad esempio, il concetto di "mollica" è lessicalizzato in Italiano, ma non in Inglese, Bentivogli et al. 2000): in generale la possibilità di allineamento tra i vari wordnet in lingue diverse si basa sull'ipotesi che la struttura concettuale sia effettivamente universale anche se gap linguistici, cioè concetti/synset non lessicalizzati in una lingua, porterebbero a pensare altrimenti. Guardando all'italiano in termini quantitativi, le risorse presenti in EuroWordNet ed in MultiWordNet essenzialmente si equivalgono: ItalWordnet (Roventini et al. 2000), parte di EuroWordNet, comprende 47.000 lemmi, 50.000 synsets e circa 130.000 relazioni semantiche. La parte italiana di MultiWordNet (Bentivogli et al. 2002) comprende invece circa 46.000 lemmi, quasi 39.000 synsets allineati con il WordNet inglese e circa 120.000 relazioni semantiche (i dati si riferiscono alla versione 1.42).

[WordNet e l'ambiguità semantica] Questo approccio alla struttura globale del lessico è molto interessante perché ci permette di gestire in modo soddisfacente il problema dell'*ambiguità semantica*. Ricordiamo che in 4.1.2 avevamo parlato di tre livelli di ambiguità; giunti a questo punto siamo in grado di rappresentare adeguatamente il livello di *ambiguità lessicale* (alla parola "vecchia" possiamo associare sia il PoS "aggettivo" che il PoS "nome"), quello dell'*ambiguità sintattica* (alla frase "ho visto un uomo con il cannocchiale" possiamo assegnare due strutture soggiacenti: [ho visto [un uomo [con il cannocchiale]]] Vs. [ho visto [un uomo] [con il cannocchiale]]). Il terzo livello di ambiguità, quello semantico, appunto, trova un'esplicita rappresentazione in WordNet: parole *polisemiche*, che hanno cioè significati distinti, sono associate a synset distinti in posizioni diverse della rete di relazioni di wordnet (rete che tecnicamente si chiama *grafo*).

Ad esempio prendiamo di nuovo la parola ambigua "pianta": secondo ItalWordNet questa parola ha almeno tre significati distinti che sono i seguenti:

- S.1 - nome generico di ogni vegetale arboreo, arbustaceo o erbaceo
- S.2 - parte del piede.
- S.3 - rappresentazione grafica in sezione orizzontale di un edificio, di un terreno, di un centro abitativo ecc..

Se analizziamo localmente la porzione di rete di relazioni che coinvolge questi synset, notiamo che tali nodi corrispondono a punti chiaramente distinti nel nostro grafo:

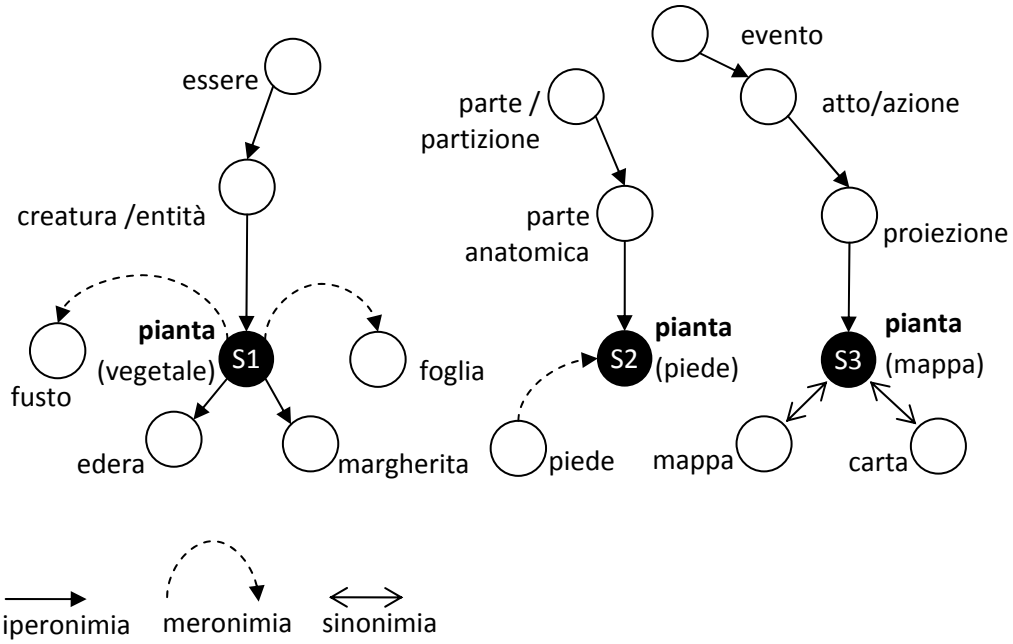


Figura 4. Espansione dei tre synset associati alla parola “pianta” (ItalWordNet)

Con questo strumento siamo quindi in grado di rappresentare adeguatamente l’ambiguità semantica assegnando synset diversi alle parole ambigue.

4.2.1.3 Il lessico come risorsa fondamentale per la descrizione linguistica

Per concludere questa riflessione sul lessico, riassumiamo alcune proprietà interessanti che siamo riusciti a descrivere esplicitamente: prima cosa, abbiamo visto che la nostra risorsa lessicale deve essere adeguatamente strutturata sia a livello di parole singole (lemma, flessione, tratti fonetici, morfo-sintattici ecc.) che a livello di struttura globale (organizzazione globale delle relazioni tra le parole, dal semplice ordinamento alfabetico, a quello per frequenze fino alla rappresentazione delle relazioni semantiche significative). In questo senso, il lessico è una fondamentale fonte di informazione morfo-sintattica oltre a poter divenire un agile strumento per la rappresentazione di domini di conoscenza (in questa accezione viene talvolta definito *ontologia*).

Un lessico può essere valutato su diverse scale: ad esempio la *copertura*, che esprime il numero di termini presenti nel lessico rispetto a quelli teoricamente necessari a rappresentare un determinato ambito; questo dato ci fornisce la misura dell’adeguatezza rispetto al dominio linguistico che si vuole rappresentare. La copertura può essere misurata non solo in estensione (numero di termini), ma anche in ricchezza (numero di tratti associati al lessico).

Un ulteriore parametro di valutazione del lessico è la sua *estensibilità*, ovvero una misura della facilità con cui si riescono ad integrare nuove informazioni morfo-sintattiche o semantiche e nuovi termini. In genere la completezza in termini di copertura e la facilità di espansione del lessico non garantiscono né la plausibilità psicolinguistica, né l’efficienza computazionale (vedi *full listing hypothesis*). In ultima analisi saranno i domini applicativi in cui i lessici verranno utilizzati a determinarne l’efficienza e l’utilità delle soluzioni adottate.

A livello astratto l’informazione lessicale può essere facilmente rappresentabile sia in forma tabellare (o di database) che in formato XML. La tabella di seguito rappresenta nei due formati le informazioni *ortografiche*, *fonetiche*, i tratti *morfologici* e *sintattici* e gli indici semantici

database		XML
token:	viole	<lex lemma=“viola" pos=”nome”
lemma:	viola	subpos=”comune.cosa” num="s" gen="f"
trascrizione		sem="3">
fonetica:	/vjɔle/	viole
categoria:	nome	</lex>
sottocategoria:	comune, cosa	
accordo di numero:	plurale	
accordo di genere:	femminile	
semantica:	synset n. 3 (ItalWordNet)	

Tabella 10. Entrata lessicale per la parola pioggia in formato tabella di database e XML.

4.2.2. La rappresentazione statistica dell’informazione all’interno di un testo

Riprendiamo adesso in esame il breve testo di Calvino riportato in (40). Nella sezione precedente abbiamo visto come un testo del genere possa essere descritto a diversi livelli in modo da renderlo fonte esplicita di informazione linguistica che la macchina può utilizzare: ognuna delle 100 forme distinte presenti in (40) può in effetti essere arricchita di informazioni quali la pronuncia, la categoria grammaticale (PoS), i vari tratti morfo-sintattici ad essa associati, il significato a cui rimanda ecc. Queste informazioni esplicite vengono solitamente aggiunte manualmente. Ciò garantisce un affidabile risultato ma ha spesso costi di sviluppo e mantenimento notevoli, costringendo ad un tedioso e lungo lavoro di etichettatura, correzione ed aggiornamento collaboratori linguistici con competenze avanzate (sia linguistiche che informatiche).

Un supporto a questo tipo di arricchimento dei dati arriva dall’analisi quantitativa delle informazioni che implicitamente sono racchiuse anche in un testo grezzo: la prima informazione implicita di cui abbiamo già parlato è la *frequenza* delle parole; anche in un testo brevissimo come quello trascritto nell’esempio in (40) si possono osservare alcune cose interessanti: le parole a più alta frequenza (“a” con 8 occorrenze, “in” con 5, “di” con 4, “il” con 3, “un” 3…) sono tutte *parole funzionali* (4.1.1.2), mentre gran parte delle parole che occorrono una sola volta (che per il fatto di occorrere una sola volta nella collezione in analisi si chiamano *hapax*) sono *parole contenuto*, cioè nomi, verbi e aggettivi (“venir”, “riusciva”, “marasma”, “abituali”...). Se poi le parole presenti nel nostro testo di riferimento non sono poco più un centinaio ma diversi milioni, si possono iniziare anche a fare previsioni più raffinate sulla tipologia delle classi di appartenenza (ad esempio morfo-sintattiche) rispetto alla frequenza dei vari token e alla loro distribuzione. Iniziamo a discutere la segmentazione del testo a basso livello (parole e lemmi) per poi passare all’analisi di informazioni quantitative che ci potranno dire qualcosa di utile sul testo che stiamo analizzando.

4.2.3.1. Tokenizzazione, normalizzazione e lemmatizzazione

[Tokenizzazione] Suddividere un testo in parole può sembrare un’operazione banale, ma basta concentrarsi su alcuni semplici aspetti “tipografici” per capire che anche questo processo, definito di *tokenizzazione*, può essere abbastanza complesso. Prendiamo un breve testo esemplificativo:

(41) Il sig. P. Pallino rappresentato e difeso dall'avv. Mario Rossi, notifica, ex-art.150 C.P.C., agli eredi e/od aventi causa di Gianni Bianchi, nato a Castelnuovo V.C. (PI) il 1° aprile 1908 e deceduto in Sassari (SA) l’11 aprile 2008, presso il Tribunale di Blablabla Sez. Distaccata, l'atto di sostituzione della locuzione 'Figura in Catasto alla partita terreni 953, foglio XI, mappale 335, are 1,59' con la seguente locuzione: 'figura in Catasto alla partita terreni 953, fogli X-XI, mappale 325, are 00,96'.

L’ipotesi che le parole siano semplicemente separate da spazi e/o segni d’interpunzione si scontra con l’evidenza di abbreviazioni del tipo “sig.” e “avv.” che non dovrebbero essere ricondotte alle forme “sig”

“avv” che non sono parole dell’italiano standard, ma piuttosto, rispettivamente, a “signore” e “avvocato”. Anche i punti che separano “C.P.C.” o “V.C.” non segnalano esattamente la fine di una frase e/o il confine di una parola, ma sono il risultato di un’abbreviazione che in alcuni casi può essere facilmente risolta con conoscenze comuni (“C.P.C.” sta per “Codice di Procedura Civile”, “V.C.” per “Val di Cecina”) e in altri casi deve essere mantenuta come tale (“P.” può ragionevolmente essere l’abbreviazione di “Pinco”, ma anche di “Paolo”) **[Text Normalization, TN]** Questo primo problema è noto come *normalizzazione del testo* (*Text Normalization*, *TN*); considerato talvolta ortogonale, talvolta propedeutico alla tokenizzazione, questo compito richiede chiaramente regole specifiche per ogni lingua ed è spesso reso complesso da varie difficoltà “contestuali”: ad esempio nei contesti “sig. Franca”, “sig. Azeglio” o “sig. Gianni” l’abbreviazione “sig.” non dovrebbe essere normalizzata allo stesso modo, ma “signora” nel primo caso, “signor” nel secondo e “signore” nel terzo. Simili difficoltà riguardano la disambiguazione di sigle come “SS.” in contesti come “SS. Giovanni e Paolo” (santissimi Giovani e Paolo) Vs. “SS. 345” (strada statale 345). La normalizzazione è nel complesso un passo cruciale, oltre che per disambiguare parole che potrebbero venire mal segmentate, nei task di *Text-To-Speech*, in particolare per quanto riguarda la traduzione in testo di date (1° Aprile 09, 1 Maggio 1998, I Gennaio ’99), numeri telefonici (ad esempio “333-3216548” che deve essere letto/riconosciuto come “tre tre tre, tre due uno sei cinque quattro otto” e non “trecentotrentatre tre milioni e duecentosedicimila cinquecentoquarantotto”), indirizzi web/mail (www.google.com, va riconosciuto come indirizzo web e non segmentato in tre distinte parole “www” “google” “com”) e così via. Per completezza, segnaliamo che il processo inverso di normalizzazione, cioè quello che parte dal riconoscimento del parlato e deve trascrivere in testo una sequenza di suoni (ad esempio “primo aprile novantasette” ci aspettiamo che possa essere correttamente trascritto come “01/04/1997”) si chiama *Inverse Text Normalization* (*ITN*).

[Una parola o molte parole?] Tornando alla nostra tokenizzazione, abbiamo compreso che non tutti i punti separano token distinti o completi così come non tutte le virgole (“Rossi, notifica” Vs. “1,59”). Inoltre vari caratteri, come gli apici (o i trattini e le barre), certe volte vanno considerati indicatori di elisione (“l'atto”), e quindi attaccati al token che li precede (talvolta anche a quello che lo segue “'st'apostrofo”), altre volte semplici virgolette (“'Figura…'”), e quindi semplicemente rimossi dai token che li precedono e seguono. Altre volte ancora potrebbero indicare indicazioni temporali (“ha corso 2000 metri in 6' e 53'”). Questo ci porta a riflettere sul fatto che alcuni token che sono rappresentati in modo ortograficamente identico in realtà, a seconda del contesto, rappresentano cose/parole diverse; l’altra faccia del problema è che ci sono forme ortograficamente rappresentate come distinte (“11 Aprile”, “foglio XI”) che in realtà sottendono la stessa parola (“undici”), anche se questo è talvolta solo un accidente contestuale (in “XI secolo” “XI” corrisponde al numerale “undicesimo”, non al cardinale “undici” come nei casi precedenti). Più banale, ma comunque degno di essere sottolineato, è il problema delle maiuscole: “figura” e “Figura” sono due token distinti per la macchina (come abbiamo visto nel capitolo 1 “F” e “f” hanno due codifiche ASCII o UTF-8 distinte e quindi le due parole sono diverse per il computer). Tutti questi problemi estendono notevolmente quello che inizialmente avevamo cercato di circoscrivere come problema di tokenizzazione; dall’includere riflessioni sulla normalizzazione di abbreviazione o caratteri speciali in questo task, a parlare di *lemmatizzazione* il passo è breve. **[Lemmatizzazione]** Dipendentemente dal tipo di analisi quantitative a cui uno vuol sottoporre il proprio testo, riconoscere “foglio” e “fogli” come due token collegati allo stesso “lemma”, cioè alla stessa forma che ci aspettiamo di trovare in un dizionario standard dell’Italiano (nel caso dei verbi ci aspettiamo di trovare una forma finita, nel caso di nomi/aggettivi una forma maschile e singolare), potrebbe essere una cosa molto utile. Il processo di estrazione/annotazione del lemma dalla forma flessa si chiama appunto *lemmatizzazione*. Tale processo può essere svolto (semi)automaticamente utilizzando macchine a stati finiti del tipo di quelle descritte in Figura 3, appunto adatte a rappresentare un

lessico come composto da radici e flessioni e quindi potenzialmente utilizzabile per ricavare la forma di default dal token flesso. Un noto esempio di "lemmatizzatore" per l'Inglese è il *Porter Stemming* (Porter 1980) che permette di estrarre la *radice* (*stem*) dalla forma flessa attraverso l'applicazione di semplici regole di riscrittura del tipo cancellazione della flessione "-ing" (-"singing" > "sing"). In Italiano queste potrebbero essere ipotetiche regole di lemmatizzazione:

- -ava > -are ("mangiava" > "mangiare")
- -ando > -are ("mangiando" > "mangiare")
- -ato > -are ("mangiato" > "mangiare")

Facile comunque trovare eccezioni che rendono queste regole troppo "potenti" e quindi inadeguate in vari contesti ("ignava" > *"ignare", "quando" > *"quare", "palato" > *"palare").

4.2.3.2. Concordanze, frequenze e contesti d'uso

La lemmatizzazione solleva un'importante problema già discusso in 4.1.2: parole diverse possono corrispondere a lemmi identici; d'altra parte abbiamo anche visto come parole identiche possano essere "ambigue", cioè possano dover essere normalizzate in modo diverso ("foglio XI", "XI secolo"); oppure possano significare cose diverse, sia a livello semantico ("pianta vegetale", "pianta del piede") che a livello lessicale ("vecchia" nome, "vecchia" aggettivo). L'analisi quantitativa della distribuzione di queste parole all'interno di una collezione di testi e l'estrazione del contesto in cui esse compaiono può essere d'aiuto per rispondere a varie domande quali la frequenza con cui si associa un certo tratto a queste parole a seconda del contesto (in casi di ambiguità lessicale e/o semantica) e/o la co-occorrenza di specifici termini con specifici tratti.

[L'Index Thomisticus e le concordanze] Storicamente questo tipo di domande ha giustificato le prime ricerche che a partire dagli anni '50 hanno gettato le basi della linguistica computazionale ponendosi il problema di strutturare in formato digitale testi di vario genere per estrarne indici di frequenza, dizionari di macchina e recuperare efficientemente espressioni linguistiche attraverso interrogazioni mirate. Padre Roberto Busa S. J., presso il Centro per l'Automazione dell'Analisi Linguistica (CAAL) di Gallarate, è stato il creatore del famoso *Index Thomisticus*, un archivio digitale di testi composti da circa 10 milioni di parole (una dimensione notevole per la potenza di calcolo delle macchine dell'epoca) che raccoglie tutte le opere di Tommaso D'Aquino, corredato da un sistema per l'estrazione delle *concordanze*, cioè le occorrenze di parole chiave nel loro contesto (Busa 1980). Un formato standard per la visualizzazione delle concordanze è il *KeyWord in Context* (*KWIC*) che riporta in colonna centrale la parola chiave cercata e a destra e a sinistra un intorno di un numero definito di caratteri che rappresenta il contesto di occorrenza della parola ricercata (nel caso in esempio la parola "in" in un contesto di 40 caratteri):

(42)

esattezza: contare oggetti, ordinarli	*in*	figure geometriche, risolvere problemi
ambiguo, anch' egli si sentiva annegare	*in*	questa morbida penombra, non riusciva p
io. stava male: erano quelli i momenti	*in*	cui si sentiva venir meno; alle volte s
iasi cosa avesse davanti. o a metterle	*in*	fila, a ordinarle in quadrati o piramid
nti. o a metterle in fila, a ordinarle	*in*	quadrati o piramidi. l' applicarsi a q

All'inizio di ogni riga possono essere riportate le coordinate della posizione della parola chiave all'interno del testo originale (e.g. parola 12, riga 34). Per velocizzare la ricerca delle parole, tali coordinate possono essere estratte off-line creando quello che si definisce un *inverted index*, cioè un indice di tutte le occorrenze della parola all'interno del testo, e la loro frequenza:

parola	frequenza	posizione (riga:carattere)
casa	12	9:45, 22:21, 122:11, 124:53, 129:3, 145:21, 167:22, 232:14, 235:20, 267:43, 322: 3, 345:27
cane	7	14:1, 22:12, 35:32, 145:13, 223:17, 321:31, 345:16
frequente	1	229:10

Tabella 11. Esempio di Inverted Index.

L'utilità di isolare rapidamente le parole ricercate con il loro contesto è evidente: la stessa parola può essere usata con funzioni diverse (nell'esempio la preposizione "in" può introdurre sia sintagmi che esprimono il luogo entro cui si svolge l'evento, es. "*in* questa morbida penombra", che il modo in cui si svolge "mettere *in* fila") una tale lista ordinata permette di calcolare rapidamente le frequenze di un uso rispetto ad un altro e la prototipicità di certi marcatori morfologici che ne determinano la funzione.

[Snippet] Un recente impiego di algoritmi che estraggono parole chiave isolando il loro contesto è la formazione degli *snippet* dei motori di ricerca (laddove non sussistano descrizioni soddisfacenti del sito che viene identificato come rilevante rispetto alla parola chiave cercata):

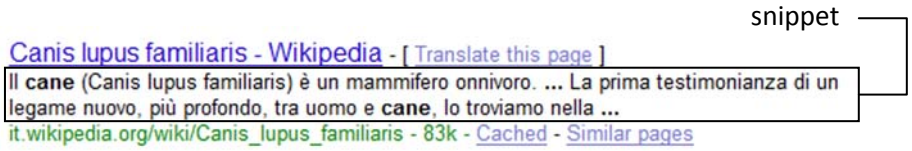


Figura 11. Snippet da Google prodotto dalla ricerca della parola "cane".

Tale compito presenta chiaramente varie altre criticità rispetto all'estrazione delle concordanze: ad esempio il contesto non può essere definito in termini di numero di caratteri visto che le espressioni isolate non solo devono comprendere parole non troncate, ma devono, per quanto possibile, isolare frasi complete o comunque comprensibili. L'algoritmo utilizzato dai vari motori di ricerca per l'estrazione di questi contesti in cui occorre la parola chiave cercata è ignoto alla comunità scientifica, ma il task in generale è molto studiato all'interno di quella branca della linguistica computazionale che prende il nome di *Text Summarization* (riassunto automatico del testo). Quello che si cerca di fare in questo ambito non è generare un vero e proprio riassunto della pagina, ma isolare le porzioni di testo in cui compaiono le parole con i migliori indizi statistici (parole non funzionali a più alta frequenza all'interno del testo/collezione di testi individuati). Anche in questo caso l'informazione quantitativa (numero di occorrenze di certe parole e la loro co-distribuzione in un determinato intorno di testo) e qualitativa (parole contenuto, non funzionali) risulta rilevante per risolvere un compito complesso come questo.

4.2.3. I corpora linguistici

Dovremmo essere a questo punto consapevoli del fatto che un testo, e quindi la lingua in esso rappresentata, può essere descritto sia in termini qualitativi (ad esempio categorie morfosintattiche, lessici computazionali e grammatiche a struttura sintagmatica) che quantitativi (indici di frequenza, collocazioni ecc.). In entrambi i casi un importante supporto alla ricerca è dato dalle banchi dati che custodiscono campioni linguistici, annotati e non. Queste banche dati a cui ci siamo a vario modo riferiti in queste pagine sono i *corpora*: collezioni necessariamente finite di testi raccolte con criteri sistematici al fine di rappresentare la reale distribuzione di un dato linguistico all'interno di un certo ambito ecologico/naturale. I criteri per la selezione, la raccolta e la campionatura del materiale, il suo formato, le fonti, i periodi di campionamento ecc. sono tutti criteri fondamentali per rendere un corpus *bilanciato* e attendibile in termini quantitativi e qualitativi. Spesso la *corpus linguistics*, quella branca della linguistica computazionale

che appunto studia questi archivi di informazione linguistica, è stata criticata dal fronte Chomskyano/generativista poiché la nozione di frequenza oscura in vario modo l'intuizione scientifica che soggiace alla grammaticalità di una frase e che con buona approssimazione abbiamo sottolineato nel paragrafo 4.1.1: un corpus per sua natura sarà sempre finito, contrariamente all'insieme delle espressioni di una qualsiasi lingua naturale; parlare quindi di frequenza può portare alla marginalizzazione di dati linguistici che in realtà sono perfettamente grammaticali e che permettono di evincere cruciali proprietà linguistiche che altrimenti sarebbero imperscrutabili nei contesti ecologici di produzione spontanea, un po' come gli esperimenti di laboratorio in una qualsiasi disciplina scientifica sopperiscono alla carenza di dati particolari o di contesti estremi necessari per avvalorare/confutare una teoria.

Ciò detto, è importante apprezzare come i corpora rappresentino sistemi informativi fondamentali per rappresentare dati linguistici quantitativamente e qualitativamente significativi per implementare *sistemi esperti* (ad esempio le registrazioni telefoniche nei call center e negli help desk servono a compilare le domande richieste più di frequente, *Frequently Asked Questions*, *FAQ*) o per l'estrazione di grammatiche (corpora annotati morfo-sintatticamente permettono di estrarre automaticamente le regole di riscrittura ed il lessico che formano le grammatiche a struttura sintagmatica). La psicolinguistica inoltre, e più in generale lo studio della performance linguistica, non può prescindere dalle analisi che prendono in esame l'effettiva distribuzione dei dati linguistici (corpora di produzioni di bambini che acquisiscono la prima o la seconda lingua, parlanti bilingui, soggetti con disturbi linguistici specifici, *Specific Language Impairment*, *SLI*, sordi, afasici ecc. sono rappresentativi di produzioni per cui è importante avere informazioni quantitative sulla reale distribuzione dei più svariati fenomeni linguistici).

In generale possiamo effettuare una rozza bipartizione tra corpora *non strutturati*, cioè banche dati in cui l'unica informazione linguistica presente è quella testuale (ad esempio file di testo con al più formattazione grafica come titolatura, colonne, giustificazione ecc.) e corpora *strutturati* (convenzioni precise indicano la natura dei dati linguistici, quali i PoS tag e l'indicazione della struttura dei costituenti e/o delle dipendenze).

Prima di passare all'uso che si può fare di questi strumenti, vediamo brevemente alcuni importanti corpora dell'una e dell'altra specie.

4.2.3.1. Brown corpus

Il *Brown corpus* (Francis and Kucera, 1964, Brown University) è uno dei primi corpus redatti con scrupolo scientifico ed è considerato un punto di riferimento per tutta la corpus linguistics, oltre ad essere rappresentativo della lingua americana degli anni '60. Il corpus è *sincrono*, cioè composto da testi prodotti in un particolare periodo storico (1961), e *bilanciato*, cioè fortemente rappresentativo di un particolare contesto linguistico, quello del linguaggio giornalistico, per cui si è scelto di selezionare articoli di diversa tipologia in modo da rispecchiare la reale distribuzione di genere, quantità e qualità delle espressioni linguistiche effettivamente presenti nella stampa americana dell'epoca.
Complessivamente il corpus raccoglie circa un milione di parole racchiuse in 500 testi (di circa 2000 parole ciascuno) che sono suddivisi nelle seguenti 15 categorie (il carattere ad inizio riga indica il codice di riferimento alla categoria usato nel corpus):
A. stampa: reportage (44 testi)
B. stampa: editoriali (27 testi)
C. stampa: periodici (17 testi)
D. religione (17 testi)
E. arti e mestieri (36 testi)
F. tradizioni popolari (48 testi)

Inizialmente il corpus si presentava non-strutturato in formato testo semplice (i codici iniziali indicano la categoria del testo, il numero identificativo di tale testo e la riga a cui l'espressione si colloca):

(43) *A01 0010 The Fulton County Grand Jury said Friday an investigation*
A01 0020 of Atlanta's recent primary election produced "no evidence" that
A01 0030 any irregularities took place. The jury further said in term-end
A01 0040 presentments that the City Executive Committee, which had over-all
A01 0050 charge of the election, "deserves the praise and thanks of the
A01 0060 City of Atlanta" for the manner in which the election was conducted.

Successivamente è stato annotato anche a livello morfosintattico, ma anche nella sua scarna veste testuale, ha stimolato in vario modo la ricerca linguistica permettendo di produrre statistiche interessanti sulla frequenza delle parole in inglese, confermando ad esempio che le parole a più alta frequenza sono quelle funzionali ("the", "of" ecc.). Si è osservato inoltre che la frequenza delle parole decresce asintoticamente e che una "lunga coda" di hapax (parole con una sola occorrenza) rappresenta circa la metà dell'intero vocabolario estraibile da questo corpus (circa 50.000 parole).

4.2.3.1. Childes

Il database Childes (acronimo di Child Language Data Exchange System, MacWhinney & Snow, 1985, Carnegie Mellon) è un corpus semi-strutturato, cioè non completamente annotato morfo-sintatticamente, ma comunque arricchito di varie informazioni linguistiche quali i contesti di produzione di certe espressioni, informazioni anagrafiche sui parlanti e su certe proprietà morfologiche delle parole. Nel suo complesso si presenta come un archivio di trascrizioni di produzioni spontanee di bambini (solitamente dai 14 mesi ai quattro anni di età) che interagiscono con adulti in varie situazioni. Generalmente ogni trascrizione si riferisce ad una conversazione di durata variabile dai 20 ai 60 minuti. Childes è in realtà composto da innumerevoli corpora, in lingue diverse, di soggetti con caratteristiche diverse sia per tipologia (bambini che apprendono la prima lingua, bambini bilingui, bambini con disturbi specifici del linguaggio, sordi ecc.) che per metodologia (registrazione diacronica dello stesso soggetto per diversi anni della proprio vita, registrazione sincronica di diversi soggetti in un particolare momento ecc.). In comune, questi corpora hanno la codifica delle trascrizioni che è definita secondo le specifiche del formato *CHAT* (*Codes for the Human Analysis of Transcripts*). Un software specifico permettere di trascrivere i file in questo formato e di estrarne informazioni rilevanti rispetto a determinati parametri linguistici (co-occorrenze di forme speciali, ricerca per errori di spelling ecc.). Tale programma si chiama CLAN (Computerized Language ANalysis) e permette di calcolare indici molto importanti come la lunghezza media degli enunciati (MLU, Mean Length of Utterance) che forniscono un indice di comparazione tra produzioni di soggetti di età e tipologia diversi. Un file codificato in CHAT si presenta così:

(44) *@UTF8*
@Begin
@Participants: CHI Cam Target_Child, DON Mother
@Age of CHI: 3;4.9
@Sex of CHI: female
@Birth of CHI: 3-MAY-1988
@Date: 12-SEP-1991
**DON: quale volevi ?*

**CHI:* *io volevo questo .*
**DON:* *si ma cosa, che canzoni ci sono, sopra .*
**CHI:* *non lo so .*
**DON:* *come non lo sai ?*
[...]
@End

Nel file, oltre ad informazioni che riguardano la codifica dei caratteri, il nome del partecipanti alla conversazione, l'età del soggetto da analizzare ecc. (sempre precedute dal segno "@") si alternano i soggetti parlanti: ogni riga esprime un turno e un codice di tre caratteri preceduti dal segno "*" identificano univocamente il parlante.

I dati custoditi nei corpora di Childes hanno permesso di produrre importantissimi studi sull'acquisizione della prima lingua: per quanto riguarda l'Italiano, ad esempio, si è potuto determinare che i bambini sono sensibili alla finitezza del verbo pur usando forme verbali non-standard (Guasti 1993-94); in effetti i clitici, che in italiano devono stare in posizione *proclitica*, cioè prima del verbo, con verbi finiti ("*lo* mangio" Vs. *"mangio *lo*") e in posizione *enclitica*, cioè dopo il verbo, con verbi finiti ("mangiar-*lo*" Vs. *"*lo* mangiare") sono sistematicamente posizionati correttamente anche da bambini molto piccoli:

(45) a. non puoi fam-*mi* questo (Diana 2 anni e 5 mesi)
b. *mi* son fatta male

4.2.4.3. British National Corpus (BNC) e la PENN treebank

Due importanti risorse che per dimensioni e qualità sono diventate il punto di riferimento della maggior parte dei ricercatori che tentano di sviluppare software per il processamento automatico del linguaggio sono il British National Corpus (BNC) (Leech 1992) e la PENN treebank (Marcus & al. , 1993). Il BNC è composto da circa 100 milioni di parole ed è un corpus *bilanciato*, *sincrono* di testi scritti (90%) di varia natura (articoli di giornale, testi scientifici ecc.) e trascrizioni del parlato (10%) (conversazioni spontanee) in inglese britannico tra il 1991 e il 1994. Tale corpus è codificato in *SGML* (standard *TEI*, *Text Encoding Initiative*) e annotato morfo-sintatticamente per PoS.

La PENN Treebank è composta invece da un milione di parole provenienti da articoli del 1989 del Wall Street Journal più un campione di materiale proveniente dalle trascrizioni di dialoghi ATIS-3 (Automatic Terminal Information Service). Lo standard di etichettatura (*Treebank II style*) è uno standard ampiamente accettato in tutta la comunità scientifica e si presenta come nell'esempio di seguito:

(46) (S (PP (IN **Of**) (NP (NN **course**))) (, ,)
(S (S (NP (DT **some**)
(PP (IN **of**) (NP (PRP$ **my**) (NN **color**) (NNS **values**))))
(AUX (VBP **do**)) (NEG (RB **not**)) (VP (VB **match)**
(NP (NP (DT **the**) (JJ **old**) (NN **Master**)) (POS **'s**))))
(CC **and**)
(S (NP (DT **the**) (NN **perspective**)) (VP (VBZ **is**) (ADJP (JJ **faulty**))))
(CC **but**)
(S (NP (PRP **I**)) (VP (VBP **believe**)
(S (NP (PRP **it**)) (AUX (TO **to**)) (VP (VB **be**)
(NP (DT **a**) (JJ **fair**) (NN **copy**))))))))

Per *treebank* si intende quindi una banca dati di alberi sintattici, in cui ogni parola è etichettata per PoS e ogni costituente annotato per categorie che ricordano molto quelle illustrate in 4.2.1.1 (che in effetti si ispirano proprio a questo pionieristico lavoro di annotazione). I codici alfabetici accanto alle parentesi tonde aperte indicano la tipologia di costituente e PoS ("NN" sta per nome comune, "VBP" per verbo tempo presente, "NP" per sintagma nominale ecc.)

Il BNC e la PENN treebank sono il banco di prova di moltissimi sistemi di processamento automatico del linguaggio; solitamente, questi corpora vengono suddivisi in due parti: la prima viene usata per il *training*, ovvero, utilizzando tecniche statistiche, si estraggono informazioni sulle regole di riscrittura (vedi par. 4.1.1) che descrivono gli alberi effettivamente presenti nella treebank e, in caso di ambiguità, le ordinano per frequenza; una seconda parte è costituisce il *test* vero e proprio su cui il sistema deve mostrare la propria performance, ovviamente senza aver mai visto prima gli alberi in esso contenuti. I risultati vengono valutati quantitativamente su varie scale (sostituzioni di etichette, asimmetrie nella marcatura di inizio/fine costituente, posizione gerarchica diversa ecc.).

4.2.4.4. Corpora dell'Italiano

Sebbene le risorse più significative in termini di impatto per la ricerca in linguistica computazionale siano state sviluppate per l'inglese, anche l'italiano presenta una discreta vitalità in questo campo: accanto a corpora non strutturati come Parole (corpus bilanciato di circa 20 milioni di parole, riferimento per l'Italiano contemporaneo, sviluppato presso l'Istituto di Linguistica Computazionale di Pisa, Marinelli et al. 2001) e "La repubblica" (Baroni et al. 2004) da cui, come abbiamo visto in 4.2.1.1. è stato estratto un interessante lessico per l'Italiano, recentemente sono state sviluppate diverse treebank annotate a molti livelli: prendiamo la Turin University Treebank (TUT, Bosco et al. 2000) come rappresentativa di questa ultima generazione di treebank. **[Costituenti e dipendenze]** La TUT si presenta come un piccolo corpus di circa 2400 frasi (tratte principalmente dal codice civile italiano da articoli di giornale) annotate morfo-sintatticamente sia per costituenti (come la PENN Treebank) che per *dipendenze*: all'interno di un costituente, gli elementi presenti (parole e/o altri costituenti) sono in relazione funzionale tra di loro; ad esempio la relazione di complementazione che sussiste tra il verbo e il suo complemento oggetto, o quella di specificazione della determinazione che sussiste tra un nome e il suo articolo. Queste informazioni sono fondamentali per effettuare varie analisi (ad esempio per disambiguare il punto di attaccamento di un sintagma preposizionale a seconda dell'elemento da cui è dipendente, vedi 4.1.2). Il *tagset* utilizzato sia a livello morfo-sintattico che a livello funzionale è discretamente ricco (Chesi et a. 2008). Per questo motivo la TUT è diventata il punto di riferimento per lo sviluppo e la valutazione dei vari software di parsing per l'Italiano (Bosco et al. 2008).

4.3. Come si usano le informazioni linguistiche

La discussione sulle proprietà linguistiche condotta in questo capitolo adesso dovrebbe permetterci di rispondere con cognizione di causa alla domanda "perché gli uomini capiscono il linguaggio naturale ed i computer no?". La risposta è legata alla necessità di creare complesse banche dati annotate con informazioni morfo-sintattiche e semantiche, all'esigenza di costruire lessici completi ed efficienti organizzati come ontologie, all'urgenza di stabilire criteri per gestire l'ambiguità ad ogni livello e la possibilità di dover assegnare un significato ad un'espressione mai vista prima. Tutto ciò è spontaneamente nelle corde di un parlante nativo che ha a disposizione competenze innate (ad esempio un supporto biologico per il processamento dei fonemi e per l'organizzazione delle informazioni linguistiche in strutture gerarchiche) e acquisite naturalmente (come il lessico e certe proprietà morfo-sintattiche specifiche di una certa lingua). A questo punto possiamo chiederci quanto queste informazioni siano sufficienti ed esplicite per "addestrare" il nostro computer a capire e/o produrre espressioni in una qualsiasi lingua naturale. E questo in parte sarebbe addirittura limitativo: dagli studi psicolinguistici abbiamo appreso che il processamento linguistico è relativamente lento ed "impreciso" per certe cose, rapidissimo ed estremamente efficiente per altre: la percezione di una parola scritta richiede diversi millisecondi, la lettura di frasi complete richiede secondi, un libro viene letto in qualche ora/giorno e difficilmente il suo contenuto viene ricordato parola per parola; recuperare un'espressione all'interno di un testo non digitalizzato può richiedere ore se non ci sono indici analitici precompilati. D'altra parte la maggior parte delle ambiguità ad ogni livello viene risolta in pochi millisecondi, spesso senza neppure prendere in considerazione strategie di analisi alternative ma inadatte al contesto. Un computer invece può permettersi di memorizzare enormi quantità di informazione testuale, recuperare precise parole o espressioni in pochi secondi, trovare un documento tra milioni che soddisfi certe chiavi di ricerca. D'altra parte nessuna macchina riesce ancora facilmente a risolvere tutti i tipi di ambiguità che abbiamo discusso in questo capitolo ed il processamento del linguaggio risulta un compito in cui efficienza computazionale e completezza dell'analisi linguistica sono variabili inversamente proporzionali. Questa tensione tra processamento linguistico umano ed artificiale ha ovviamente molte sfaccettature che vale la pena di esplorare, almeno parzialmente, in quest'ultima sezione. In questo modo forse riusciremo ad apprezzare lo sforzo che abbiamo fatto fin qui per cercare di descrivere adeguatamente la nostra lingua naturale.

4.3.1. Interrogazione di corpora e l'estrazione di informazioni linguistiche

Iniziamo con il recupero di informazioni squisitamente linguistiche. Un testo in formato digitale (i.e. un corpus, 4.2.3) che eventualmente è stato annotato ci permette di fare cose molto interessanti sia per quanto riguarda lo studio della lingua che per quanto riguarda l'estrazione di grammatiche e lessici da dare in pasto a sistemi di Information Retrieval/Extraction, traduttori automatici o sistemi esperti. Ma quali sono esattamente gli strumenti che si possono utilizzare per recuperare queste informazioni? A seconda che il nostro corpus sia strutturato (quindi annotato con informazioni morfo-sintattiche e/o semantiche) o non strutturato (quindi testo semplice) ci sono strumenti adatti ai nostri scopi.

4.3.1.1. Espressioni Regolari

Le Espressioni Regolari (*Regular Expressions, RE* o *regex*) tecnicamente sono una notazione algebrica che permette di definire esattamente insiemi di stringhe di testo. In pratica si percepisce l'esigenza di utilizzare qualcosa di simile alle espressioni regolari ogni volta che si effettua una ricerca su internet o si tenta di trovare un'espressione "complessa" in un documento testuale. Domande del tipo "voglio trovare una stringa di testo che contenga la parola 'margherita' oppure la parola 'margherite'" oppure "voglio un'espressione che contenga contemporaneamente la parola 'ciclo' e la parola 'solare', possibilmente in questo ordine" o ancora "voglio una frase che inizi per 'previsione' ma che non contenga nel suo prosieguo

la parola ‘oroscopo’”. **[Pattern di identificazione]** Tutte queste domande sono esprimibili utilizzando le RE. In cuore delle RE è il *pattern di identificazione* composto da caratteri alfanumerici (compresi segni di spaziatura e di interpunzione) e da segni speciali volti a stabilire le relazioni tra i caratteri del pattern.

Sapendo che nella sintassi delle RE i caratteri tra parentesi quadre sono in alternativa (es. “[ab]” significa “a oppure b”), che il punto significa “qualsiasi carattere” (ad esempio l’espressione “.*” cattura una qualsiasi sequenza di caratteri di arbitraria lunghezza visto che la “*” esprime la ripetizione arbitraria del carattere a cui si applica; è questo l’operatore di Kleene di cui abbiamo parlato in 4.1.1.1, Kleene è peraltro l’ideatore delle espressioni regolari), che il segno “^” indica l’inizio della frase e che “[^(…)]” indica la negazione della sequenza di caratteri all’interno delle parentesi tonde, ecco la traslazione in RE delle domande pocanzi poste:

Interrogazione in lingua naturale	**Espressione Regolare corrispondente**
“voglio trovare una stringa di testo che contenga la parola ‘margherita’ oppure la parola ‘margherite’”	margherit[ae]
“voglio un’espressione che contenga contemporaneamente la parola ‘ciclo’ e la parola ‘solare’ possibilmente in questo ordine”	ciclo.*solare
“voglio una frase che inizi per ‘previsione’ ma che non contenga nel suo prosieguo la parola ‘oroscopo’”	^previsione.*[^(oroscopo)]

Tabella 12. Esempi di traduzioni da domande in linguaggio naturale ad espressioni regolari

La sintassi delle RE prevede che i caratteri alternativi, siano o inclusi tra quadre (es. [abcd] significa “a, oppure b, oppure c, oppure d”, [a-z] significa qualsiasi carattere alfabetico minuscolo, [0-1] corrisponde a “qualsiasi carattere numerico”) o separati da “|” (pipe) dentro parentesi tonde (es. (mare|spiaggia) restituisce stringe che contengono la sequenza di caratteri ’mare’ oppure ‘spiaggia’). Le parentesi tonde identificano blocchi di caratteri che vengono sempre letti in sequenza: (abc) troverà solo espressioni del tipo “abc”, mentre [abc] individuerà ogni espressione che al suo interno conterrà una “a” oppure una “b” oppure una “c”.

I caratteri speciali, come lo spazio o l’invio a capo, sono codificati solitamente da segni speciali; ad esempio in *Perl*, un linguaggio di programmazione di alto livello, spesso usato per manipolare stringhe di testo usando espressioni regolari, lo spazio o l’invio a capo si esprimono rispettivamente con “\s” e “\n”.

Infine si possono usare i moltiplicatori, ovvero degli operatori che indicano la ripetizione di un carattere o sequenza di caratteri una o più volte (“*” un numero indefinito di vole, “+” una o più volte, “?” una o nessuna volta) e degli indicatori di posizione (*ancore*) che indicano quando si deve trovare il carattere, o la sequenza di caratteri ricercata (“^” indica l’inizio frase “$” la fine). Nel caso di moltiplicatori la stringa individuata dalla RE è quella che massimizza il numero di caratteri individuabili (le RE per questo motivo si definiscono “greedy” cioè “ingorde”: l’espressione “t.*e” preferirà “torre pendente” a “torre pendente” o a “torre pendente”).

Per fare un po’ d’esercizio, ecco la tabella con le principali regole per la costruzione di pattern di identificazione:

Espressione Regolare	**Significato; corrispondenza**	**Esempio di pattern identificato**
[Dd]uomo	i caratteri dentro le quadre sono in alternativa; Duomo oppure duomo	Il duomo è nella piazza
[a-z]	qualsiasi carattere alfabetico minuscolo	Il duomo è nella piazza
[A-Z]	qualsiasi carattere alfabetico maiuscolo	Il duomo è nella piazza
[0-1]	qualsiasi carattere numerico	Il numero 1 di vicolo dell'Oro
[^a-z]	tutto fuorché lettere minuscole	Il numero 1 di vicolo dell'Oro
sali?ta	"?" indica una o nessuna occorrenza del carattere che precede; salita oppure salta	Marco deve saltare
sal.ta	il "." accetta qualsiasi carattere	Marco saluta
bu*	"*" operatore di Kleene, b seguito da un numero imprecisato (anche nullo) di *u*	buuuuuu! oppure b!
bu+	"+" indica una o più occorrenze del carattere che precede; b seguito da un numero imprecisato (ma non nullo) di *u*	buuu!
bu{n,m}	tra parentesi graffe è specificato il numero minimo (n) e massimo (m} di elementi che precedono la parentesi aperta; b seguito da un numero di u compreso tra n ed m (se m non viene specificato il numero di *u* è uguale a n)	l'espressione "bu{1,3}" accetta bu!, buu! e buuu!
^L	"^" segna l'inizio della stringa; l'espressione restituisce quindi le sole *L* che iniziano una stringa	La casa
a$	"$" segna la fine della stringa; l'espressione restituisce quindi solo le *a* che terminano una stringa	La casa
cas(a|e|ina)	"|" è equivalente alla disgiunzione logica le parentesi restringono la portata dell'operatore, in questo caso la stringa *cas* può essere seguita o da *a* o da *e*, o da *ina*	Marco vive in un casale
*	il backslash è il simbolo di *escape*, che permette cioè di cercare come testo semplice i caratteri speciali (in questo caso l'asterisco)	A*

Tabella 13. Esempi di espressioni regolari

Le RE sono utili non solo per individuare parole o espressioni specifiche, ma per rappresentare in modo compatto insiemi di stringhe. Questo è un "problema" che avevamo già incontrato nel paragrafo 4.2.1: il nostro lessico suddiviso in radici e suffissi potrebbe essere, in modo relativamente semplice, rappresentato da opportune espressioni regolari. In effetti le espressioni regolari e le macchine a stati finiti circoscrivono lo stesso insieme di linguaggi (cioè lo stesso insieme di stringhe). Ad esempio possiamo creare espressioni regolari che isolino certe radici nominali maschili e le relative flessioni singolari e plurali:

(47) (aere|bagn|cas|don|elicotter|...)[oi]

Oppure le radici verbali e le relative flessioni del presente (vedi tabella 3):

(48) (cant|vol|googl|…)(o|i|a|iamo|ate|ano)

[Registri e sostituzioni] Infine, usando i *registri*, blocchi di memoria indicizzata in cui vengono archiviati le occorrenze che soddisfano un determinato pattern tra parentesi tonde, si possono riutilizzare le stringhe individuate per sostituirle ad altre secondo il pattern "espressione regolare di partenza/espressione regolare in cui avvengono le sostituzioni".

Ad esempio l'espressione che segue permette di costruire la forma attiva di una frase ("Gianni ha comprato la casa") partendo dalla frase passiva ("la casa è stata comprata da Gianni").

(49) la (casa|macchina) è stata comprata da (Maria|Gianni)/ \2 ha comprato la \1

"\2" corrisponde all'occorrenza individuata (ad esempio "Gianni") dal secondo blocco della prima espressione regolare (Maria|Gianni), mentre "\1" è l'occorrenza (ad esempio "macchina") che soddisfa il primo blocco (casa|macchina).

4.3.4.2. TGrep

La stessa logica delle espressioni regolari può essere applicata anche alla ricerca nei corpora annotati morfo-sintatticamente.

Tra i vari strumenti che permettono ricerche mirate volte ad isolare pattern sintattici di dominanza e o precedenza tra nodi specifici (4.1.1.2), è degno di nota lo il programma TGrep (in particolare nella sua versione TGrep2 la cui sintassi è discussa in queste pagine, Rohde 2005).

Nella tabella che segue sono espresse le convenzioni che si usano per effettuare le principali interrogazioni su banche dati annotate usando TGrep2:

Espressione	**Significato**	**Alcuni esempi di occorrenze**
A < B (e.g. NP < N)	A domina immediatamente B	(NP (D la) (N casa))
A < X N (e.g. "NP < 3 N")	B è l'X-esimo nodo figlio di A	(NP (D la) (A bella) (N casa))
A <-X B (e.g. VP <- 1 V)	B è distante X posizioni dall'ultimo nodo figlio di A	(VP (AUX ho) (V corso))
A <- B (e.g. equivalente a: VP <- 1 V)	l'ultimo nodo figlio di VP è V	(VP (AUX ho) (V mangiato))
A << B (e.g. VP << N)	A domina B	(VP (V compro) (NP (N casa)))

A <<, B (e.g. NP <<, N)	B è il primo nodo figlio di A	(NP (N fiori) (A rossi))
A <<’ B (e.g. NP <<’ N)	B è l’ultimo nodo figlio di A	(NP (A bel) (N fiore))
A . B (e.g. D . N)	A precede immediatamente B	(NP (D il) (N libro))
A .. B (e.g. D .. N)	A precede B	(NP (D il) (A bel) (N libro))
A $ B (e.g. ADV $ V)	A e B sono nodi fratelli (A $ A è falso!)	(VP (ADV non) (AUX è) (V caduto))
A $. B (e.g. AUX $ V)	A e B sono nodi fratelli e A precede immediatamente B	(VP (AUX è) (V caduto))
A $.. B (e.g. AUX $ V)	A e B sono nodi fratelli e A precede B	(VP (ADV non) (AUX è) (ADV mai) (V caduto))
/^A/ (e.g. /^NP/)	identifica ogni nodo il cui nome inizia per A	(NP-SBJ (D la) (N casa))

Tabella 13. Sintassi delle espressioni regolari

Questo strumento è fondamentale per esplorare rapidamente banche dati strutturate come la PENN treebank (4.2.4.3), permettendo di estrarne dati precisi che possono essere utili sia per verificare ipotesi linguistiche che per isolare sottoinsiemi di strutture su cui addestrare riconoscitori particolari (ad esempio riconoscitori di *Entità Nominali, Named Entities, NE*, quali nomi di persone, luoghi, entità geopolitiche e/o organizzazioni).

Per quanto riguarda le ipotesi linguistiche, ad esempio si può testare l’idea che la “pesantezza” di un sintagma nominale (cioè il fatto che questo sintagma nominale sia modificato da una frase relativa come nel caso di “[il bambino [che ho visto ieri al cinema]]”) spesso produce quello che si definisce un *Heavy-NP Shift* (*HNPS*), cioè uno spostamento del sintagma in questione verso la periferia destra della frase ((50).a Vs. (50).b):

(50) a. [$_{VP}$ ho letto [$_{NP}$ il libro] [$_{PP}$ con molta curiosità]]
b. [$_{VP}$ ho letto [$_{PP}$ con molta curiosità] [$_{NP\text{-}pesante}$ il libro che parlava del paese in cui sono nato]]

La percentuale di sintagmi nominali “pesanti” marginalizzati alla fine del costituente verbale può essere semplicemente calcolata confrontando il numero di risultati prodotti dell’espressione in (51).a (che trova tutti i sintagmi nominali modificati da relativa) con il numero dei risultati prodotti da quella in (51).b (sintagmi nominali modificati da relativa effettivamente posizionati nella periferia destra del sintagma verbale che li domina) questa espressione:

(51) a.VP << (NP < VP)

b. VP <<, (NP < VP)

4.3.2 Parlare con un computer: il test di Turing

Il famoso matematico inglese Alan Turing (ideatore dell’omonima macchina) interrogandosi sul concetto di “intelligenza artificiale”, propose (in parte provocatoriamente) un famoso test (il *test di Turing* appunto) per valutare l’intelligenza di una macchina: tre attori, due umani e una macchina devono conversare attraverso terminale (modello chat-line). Un umano è l’arbitro che deve valutare, in base alle domande che può liberamente fare via computer, chi dei due interlocutori è umano e chi è la macchina. La macchina supera il testi di Turing se viene scambiata per umano. Il test di Turing ha suscitato moltissime riflessioni filosofiche sul concetto di “macchina”, “pensiero”, “linguaggio naturale” e di “intelligenza” (Searle 1980). Cosa ci interessa in questo contesto è notare che vari sistemi “conversazionali” sono stati prodotti dopo la formulazione del test e alcuni di questi, riuscendo a manipolare in modo interessante, per quanto banale,

espressioni linguistiche, sono degni di nota. Uno dei primi e più noti sistemi conversazionali è *Eliza* (Weizenbaum, anno 1966). **[Eliza e l'operazione di sostituzione]** Eliza è un sistema conversazionale che simula il comportamento di uno psicologo rogeriano, cioè semplicemente stimola l'interlocutore umano a produrre frasi di senso compiuto su temi svariati. Il sistema ovviamente non comprende assolutamente quello che gli viene detto, ma simula un flusso dialogico coerente individuando certe espressioni chiave utilizzate dal parlante umano e innestando su di queste espressioni linguisticamente ben formate precompilate dal suo programmatore.

Ecco alcuni passaggi di possibili conversazioni con Eliza:

(52)	*Utente*:	tutti gli uomini sono uguali
	Eliza:	in che senso?
	Utente:	si lamentano sempre di qualcosa
	Eliza:	puoi farmi un esempio specifico?
	Utente:	beh, il mio ragazzo mi ha fatto venire qui
	Eliza:	il tuo ragazzo ti ha fatto venire qui
	Utente:	il mio ragazzo dice che sono sempre depressa
	Eliza:	sono spiacente di sapere che sei depressa

A questo punto della nostra riflessione, se abbiamo ben compreso l'uso delle espressioni regolari e dell'operazione di sostituzione (4.3.1), siamo anche noi in grado di scrivere regole per far parlare ELIZA come nell'esempio (52):

RE da applicare alla frase dell'interlocutore	RE di sostituzione che ELIZA proporrà come risposta
sono .* (depress[o|a]|triste)	sono spiacente di sapere che sei \1
sono tutt[i|e] (.*)	in che senso sono \1?
sempre	potresti far riferimento ad un esempio specifico?

Tabella 16. L'uso di espressioni regolari e l'operazione di sostituzione in ELIZA

I sistemi conversazionali hanno trovato un terreno molto fertile nella seconda metà degli anni '90 con l'avvento di Internet e dei sistemi di chat. Noti sono i vari *chat-bot* che affollavano le prime chat-line (ad esempio ICQ) il cui unico scopo era quello di "adescare" utenti umani per propinare loro messaggi pubblicitari. Usi più lodevoli di questi strumenti sono stati fatti anche in ambito help-desk: un discreto successo è stato ottenuto da *Alice*, chat-bot della Telecom che accompagna l'utente nel sito della compagnia, rispondendo a domande generali (FAQ) attuando un modello dialogico di chat on-line e fornendo varie istruzioni per la navigazione e/o compilazione di moduli e richieste. Recentemente Alice chat-bot è stato dotato di corpo (un avatar, cioè un modello tridimensionale di essere umano dotato di espressioni facciali e vari "tic" prossemici) e di voce (la sintesi del parlato ha raggiunto livelli di notevole chiarezza e naturalezza, in questo contesto la riflessione sull'ITN, 4.2.3.1, è particolarmente rilevante).

4.3.3. Il recupero di documenti attraverso parole chiave

Concludiamo questa riflessione sul linguaggio cercando di esplorare brevemente il campo che più d'ogni altro ha tratto beneficio dalla rappresentazione digitale e riccamente strutturata dell'informazione linguistica: la ricerca di informazioni su web. L'enorme mole di dati testuali (e non) richiede sistemi efficienti di recupero ed esplorazione documenti sia in termini di rapidità che in termini di qualità della risposta. Sebbene la maggior parte dei sistemi si basi su algoritmi statistici per calcolare la rilevanza di un

determinato documento web rispetto ad una determinata query (ad esempio il *PageRank* di Google, Brin et al. 1998), descrivere adeguatamente questi algoritmi richiederebbe una trattazione più approfondita di quella che ci possiamo concedere in queste ultime pagine; ci limiteremo quindi a fornire gli input essenziali per apprezzare l'intuizione alla base della rappresentazione delle queries e dei documenti testuali e le riflessioni linguistiche che possiamo fare dopo quello che abbiamo scoperto in questo intenso excursus linguistico-formale al fine di trattare anche con richieste e documenti ambigui.

4.3.3.1. La rappresentazione della query e del testo

Con ragionevole approssimazione, partiamo dal presupposto che ogni domanda ed ogni documento su web, possa essere rappresentato essenzialmente dalle parole che lo compongono. Seguendo una convenzione standard, in questa sezione, chiameremo *documento* ogni unità di testo indicizzata, *collezione* un insieme finito di documenti, *query* la sequenza di elementi lessicali con cui si interroga un motore di ricerca e *termine* ogni elemento lessicale che occorre in un documento/query.

[Un sacco di parole] Prescindendo da qualsiasi considerazione linguistica, un documento rappresentato dalle parole in esso contenuto è visto come "un sacco di parole" (*Bag-of-Words*, *BoW*) cioè l'informazione considerata è solo quantitativa: "il cane morde il gatto" o "il gatto morde il cane" sono espressioni che avranno esattamente lo stesso "significato" da questo punto di vista, visto che entrambe le stringhe saranno rappresentate con questo insieme di parole (l'indicizzazione esprime la loro frequenza):

(53) $\{il_2, gatto_1, morde_1, cane_1\}$.

Questa intuizione trova una sua rappresentazione formale nel modello di spazio vettoriale (Salton 1971): uno spazio vettoriale è uno spazio algebrico (cioè una struttura su cui si possono calcolare precisamente le distanze tra i vari oggetti) su cui si possono proiettare vettori, cioè oggetti complessi descrivibili in termini di un numero arbitrariamente grande di componenti che li identificano. Queste componenti non sono altro che le rappresentazioni numeriche dei nostri termini e della loro frequenza.

Ad esempio la frase in (53) potrà essere rappresentata semplicemente da un vettore ($\vec{f}$) come questo che indica per ogni parola la sua frequenza assoluta (i vettori solitamente si indicano con una freccetta sopra la lettera che li identifica):

(54) $\vec{f}$ = (2, 1, 1, 1)

In generale ogni documento in una collezione potrà essere rappresentato da un vettore, le cui componenti saranno tutti i termini che compaiono nell'intera collezione e i valori delle componenti esprimeranno la presenza (1 o più) o l'assenza (0) del determinato termine all'interno del documento (*n* è il numero totale di termini nella collezione, $\vec{d}_j$ è il j-esimo documento nella collezione, $t_{x,j}$ avrà quindi valore superiore a 1 o 0 a seconda che il termine in questione sia presente con una certa frequenza o assente nel documento $\vec{d}_j$):

(55) $\vec{d}_j = (t_{1,j}, t_{2,j} \ldots t_{n,j})$

Se vogliamo interrogare la nostra collezione, formuleremo una query ($\vec{q}_k$) che in realtà è una sequenza di termini, e che quindi potrà essere rappresentata esattamente come un qualsiasi altro documento testuale nella collezione:

(56) $\vec{q}_k = (t_{1,k}, t_{2,k} \ldots t_{n,k})$

Intuitivamente, il documento che soddisfa al meglio la nostra query è quello che condivide con questa il maggior numero possibile di termini. Più precisamente, un vettore esprime le coordinate di un punto in uno spazio multidimensionale (tante dimensioni quanti sono i termini); semplificando il problema, in modo da visualizzare i vettori su uno spazio bidimensionale, ipotizziamo che la nostra query sia "albergo Firenze", la nostra collezione sia rappresentata da due soli documenti al cui interno le due parole occorrono rispettivamente 2, 3 volte e 4, 1 volta; ecco qua i nostri vettori e la loro proiezione spaziale:

(57) vettori

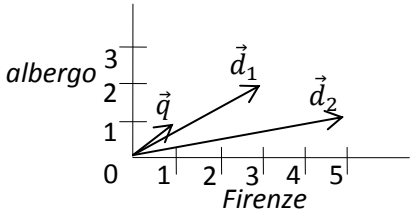
termini: (albergo, Firenze, Roma, Viareggio, … Urzulei)

$\vec{d}_1$ = (*2, 3, 0, 0 … 0*)

$\vec{d}_2$ = (*1, 5, 0, 0 … 0*)

$\vec{q}$ = (*1, 1, 0, 0 … 0*)

rappresentazione bidimensionale



In una struttura algebrica la distanza tra due vettori può essere precisamente definita (prodotto scalare) e il risultato numerico che esprime la distanza tra un documento $\vec{d}_1$ e una query $\vec{q}$ può essere semplicemente comparato alla distanza tra $\vec{d}_2$ e $\vec{q}$ (questa misura è principalmente in funzione dell'angolo sotteso tra due vettori). Al di là di come questo numero venga calcolato esattamente, quello che ci interessa comprendere è in fondo abbastanza semplice: sebbene questo metodo di confronto ci garantisca un risultato certo ed intuitivamente utile, la mera frequenza dei termini all'interno di un documento (per quanto pesata e calibrata in modo da ridurre al meglio aberrazioni statistiche), ed in generale la concezione di un testo come un "sacco di parole", sminuisce di molto il contenuto linguistico all'interno del documento e crea qualche problema in diversi scenari: se l'obiettivo della nostra ricerca è trovare un alloggio a Firenze, magari anche un documento in cui occorre spesso il termine "hotel" potrebbe essere rilevante; d'altra parte un'espressione come "non conosco nessun albergo a Firenze" contenuta in un documento qualsiasi potrebbe far balzare in cima alla classifica delle risposte migliori il documento che la contiene e che non risponde minimamente all'intento della nostra ricerca. Inoltre, come abbiamo visto osservando la distribuzione di parole ed altre frequenze su corpora di grandi dimensioni, i termini più frequenti sono quelli non portatori di contenuto referenziale evidente: gli elementi funzionali come gli articoli o le preposizioni, riccamente presenti in ogni testo, se presenti nella nostra query e se utilizzati per creare i vettori che rappresentano documenti ed interrogazioni, falseranno pesantemente i risultati della nostra ricerca. Infine se in un testo occorre molte volte una parola ambigua, ad esempio la parola "era", con la rappresentazione dei documenti che abbiamo qua descritto, non possiamo avere la certezza che il significato di quella parola sia quello che ci interessa (ad esempio "era geologica" e non "era" terza persona singolare, imperfetto indicativo, del verbo essere).

4.3.3.2. Alcune riflessioni conclusive sulla "distanza" tra testo e query

In conclusione abbiamo visto che gli indici di frequenza, cioè quante volte un termine compare in un testo, in una collezione (o corpus), sono informazioni che possiamo utilizzare in modo abbastanza efficace per rappresentare tale testo, per ricavarne una descrizione sommaria del suo contenuto, che, seppure rudimentale, così come la abbiamo descritta, ci permette comunque di effettuare ricerche mirate e di ordinare in qualche modo i risultati di tali ricerche.

[Stop list] Ragionando in modo un po' più euristico anche solo sulla frequenza, possiamo inoltre dedurre che i termini che occorrono molto frequentemente in tutti i documenti di tutta la collezione, ad esempio le parole funzionali, saranno in realtà abbastanza poco significativi e quindi dovranno essere esclusi dalla

rappresentazione dei documenti e delle queries. In effetti, quello che molti motori di ricerca fanno è compilare una lista di termini ad altissima frequenza che vengono eliminati da tutte le rappresentazioni. Questa lista, in gergo tecnico, si chiama *stop list*. **[Importanza]** L'altra faccia della medaglia è che parole che occorrono frequentemente in un solo documento, risultando relativamente infrequenti nell'intera collezione, caratterizzeranno notevolmente quel documento e saranno probabilmente significative (*importanti*) per dedurne il contenuto.

Adesso che sappiamo un po' come funziona la nostra lingua e come le varie proprietà si rappresentano in modo comprensibile anche per il nostro computer, possiamo riflettere su come migliorare la nostra strategia di IR.

[Espandere automaticamente una query] Come prima cosa, possiamo pensare che se ci interessa un albergo a Firenze, probabilmente anche documenti in cui troviamo la parola "alberghi", "hotel", "pensione" e "pensioni" saranno rilevanti. Sappiamo adesso che per espandere (automaticamente) la nostra query con le versioni plurali dei nostri termini dobbiamo far fare alla nostra macchina lo *stemming* (4.2.3.1) e, avendo a disposizione un lessico che rappresenti le parole come combinazione di radici e suffissi (4.2.1), flettere al plurale le radici nominali nella query utilizzando le adeguate transizioni nelle macchine a stati finiti.

Per utilizzare invece sinonimi, dovremmo istruire la nostra macchina ad usare strumenti come WordNet. **[Copertura e precisione]** Isolato uno specifico *synset* (4.2.1.2), potremmo espandere la nostra query con *sinonimi*, *iperonimi* ed eventualmente *iponimi*: mettendo le nostre chiavi di ricerca in congiunzione (chiedendo quindi di ricevere in risposta documenti che comprendano tutti i termini ricercati) il rischio è di aumentare la *precisione* della nostra selezione documentale (cioè quasi tutti i documenti individuati saranno coerenti con il significato del nostro termine di partenza; se tutti i documenti recuperati sono rilevanti si ottiene il 100% di precisione della ricerca), ma ridurre la *copertura* (solo un sottoinsieme dei documenti effettivamente rilevanti presenti nella collezione saranno individuati aggiungendo termini alla ricerca; se tutti questi documenti rilevanti fossero individuati si raggiungerebbe il 100% di copertura).

[Distanze tra synset e disambiguazione] Sempre usando WordNet, potremmo raggruppare automaticamente i risultati di una ricerca o comunque disambiguare le singole parole (è proprio questa una sfida che i nuovi motori di ricerca si stanno accingendo ad affrontare: è questo il caso di *Bing*, il nuovo "Decision Engine" della Microsoft). I termini nell'intorno del lemma ricercato si troveranno sicuramente tra i synset di WordNet. Poiché la rete è un grafo ampiamente connesso, è molto probabile che si riesca a trovare un "percorso" che lega due synset qualsiasi (ad esempio "gatto" e "cane" hanno entrambi come iperonimo "mammifero") che corrispondono a parole nell'intorno isolato ("il cane è un mammifero..."); in questo contesto io riuscirò a capire che quando all'interno di un documento alcuni termini che corrispondono a synset connessi strettamente con la parola in esame sono presenti, quella parola chiave, seppur polisemica, potrà essere ragionevolmente disambiguata identificando il synset ad essa associato che minimizza (sempre in termini di numero di relazioni che sono necessarie per collegare un termine all'altro) la distanza con gli altri synset individuati.

Infine anche informazioni sulla struttura morfo-sintattica ci aiutano a risolvere sia l'ambiguità lessicale che quella sintattica. Data una treebank, si posso estrarre rappresentazioni vettoriali di un testo che non solo includono la lista dei termini, ma anche, ad esempio, il PoS tag (termine, tag riferito al termine):

(58) $\vec{f}_1$ = (*Marco, N, lava, V, i, D, piatti, N, con, P, il, D, detersivo, N*)

questo ci permette di capire che quando il termine “piatti” è circondato da parole come “lava” o “detersivo” non è solitamente un aggettivo e quindi non si riferisce ad una proprietà geometrica di un oggetto, ma ad un utensile da cucina.

Bibliografia

Attardi, G. e M. Simi (2009). *EVALITA 2009: Italian Part-Of-Speech Tagging Evaluation, Task Guidelines*. FBK, Trento.

Baroni, M., Bernardini, S., Comastri, F., Piccioni, L., Volpi, A., Aston, G, Mazzoleni, M. (2004) *Introducing the "la Repubblica" corpus: A large, annotated, TEI(XML)-compliant corpus of newspaper Italian*. In Proceedings of the Fourth Language Resources and Evaluation Conference, (Lisbon: ELDA), 1771-1774.

Bentivogli, L., E. Pianta, F. Pianesi. (2000). *Coping with lexical gaps when building aligned multilingual wordnets*. In Proceedings of the Second International Conference on Language Resources and Evaluation (LREC 2000), Athens, Greece, 31 May- 2 June 2000.

Bentivogli, L., Pianta, E., & Girardi, C. (2002). *Multiwordnet: developing an aligned multilingual database*. In First International Conference on Global WordNet, Mysore, India.

Bosco, C., Lombardo, V., Vassallo, D., & Lesmo, L. (2000). Building a treebank for Italian: a data-driven annotation schema. In *Proceedings of the Second International Conference on Language Resources and Evaluation LREC* (pag.. 99–106).

Bosco, C., Mazzei, A., Lombardo, V., Attardi, G., Corazza, A., Lavelli, A., et al. (2008). Comparing Italian parsers on a common treebank: the EVALITA experience. *Proc. LREC'08*.

Brill, E. and Resnik, P. 1994. *A Rule-based Prepositional Phrase Attachment Disambiguation*. In Proceedings of the 15th conference on Computational linguistics - Volume 2, Kyoto, Japan.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7), 107-117.

Busa, R. (1980). *The annals of humanities computing: the Index Thomisticus*. Language Resources and Evaluation 14:83-90.

Chesi C., G. Lebani e M. Pallottino (2008). A Bilingual Treebank (ITA-LIS) suitable for Machine Translation: what Cartography and Minimalism teach us. Studies in Linguistics, StiL, 2:165-185

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 2:113-124.

Chomsky, N. (1957). *Syntactic structures*. Mouton, The Hague.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. MIT press. Cambridge, MA.

Eros Zanchetta and Marco Baroni (2005) *Morph-it! A free corpus-based morphological resource for the Italian language*. In proceedings of Corpus Linguistics 2005, University of Birmingham, Birmingham, UK.

Fellbaum, C. (1998). *WordNet: An electronic lexical database*. MIT press. Cambridge, MA.

Francis, W. N., & Kucera, H. (1967). *Computational analysis of present-day American English*. . Providence, RI: Brown University Press.

Frazier, L. e J. D. Fodor (1978). The sausage machine: A new two-stage parsing model. *Cognition* 6, 291–325.

Fromkin, V. A., Ratner N. B. (1998). Speech production. In Gleason J. B. e Ratner N. B. ed. *Psycholinguistics*. Harcourt Brace College Publishers, Orlando, FL.

Guasti, M. T. (1993). Verb syntax in Italian child grammar: Finite and nonfinite verbs. *Language Acquisition*, 3(1), 1-40.

Jurafsky, D., & Martin, J. H. (2008). *Speech and language processing*. An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Prentice Hall.

Koskenniemi, K. (1984). *A general computational model for word-form recognition and production.* In Proceedings of the 10th International Conference on Computational Linguistics and 22nd annual meeting on Association for Computational Linguistics. ACL, 178-181 Morristown, NJ, USA.

Leech, G. (1992). 100 million words of English: the British National Corpus. *Language Research*, 28(1), 1-13.

MacWhinney, B. e Snow, C. (1985). The child language data exchange system. *Journal of child language*, 12(2), 271.

Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2), 313-330.

Marinelli, R., Biagini, L., Bindi, R., Goggi, S., Monachini, M., Orsolini, P., et al. Zampolli A. (2003). Italian parole corpus. *Linguistica Computazionale, Computational Linguistics in Pisa, Special Issue*, 18, 401–421.

Monachini, M., & Calzolari, N. (1996). Synopsis and comparison of morphosyntactic phenomena encoded in lexicons and corpora. *A common proposal and applications to European languages. ILC-CNR, Pisa*.

Partee, B. H., Meulen, A. G. B., & Wall, R. E. (1990). *Mathematical methods in linguistics*. Springer.

Phillips, C., Pellathy, T., Marantz, A., Yellin, E., Wexler, K., Poeppel, D., et al. (2000). Auditory cortex accesses phonological categories: an MEG mismatch study. *Journal of Cognitive Neuroscience*, 12(6), 1038-1055.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3), 130-137.

Rohde, D. L. T. (2005). Tgrep2 user manual. *URL: http://tedlab.mit.edu/dr/Tgrep2*.

Roventini A., Alonge A., Calzolari N., Magnini B., Bertagna F. (2000). *ItalWordNet: a Large Semantic Database for Italian*. In Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000), Athens, Greece, 31 May – 2 June 2000, Volume II, Paris, The European Language Resources Association (ELRA), 783-790.

Salton, G. (1971). *The SMART retrieval system—experiments in automatic document processing*. Prentice Hall. Englewood Cliffs, NJ.

Searle, J. (1980). Minds, brains and programs. In *Behavioral and Brain Science*. 3:417-458.

*S*tork, D. G. (1997). *HAL's legacy: 2001's Computer as Dream and Reality*. MIT Press. Cambridge, MA.