13 Giugno 2006

Sintassi

Parsing

Marco Pennacchiotti

pennacchiotti@info.uniroma2.it

Tel. 06 7259 7717

Ing.dell'Informazione, *stanza P1B-03* (nuova ala Ing.Inf, primo piano)

Programma

- **Breve introduzione all'NLP**
  - Linguaggi Naturali e Linguaggi Formali
  - Complessità
- **Morfologia**
  - *Teoria:* Morfologia del Linguaggio Naturale
  - *Strumenti:* Automi e Trasduttori
  - *Analisi Morfologica:* con automi e trasduttori
- **Part of Speech Tagging**
  - *Teoria:* Le classi morfologiche
  - *Strumenti a Analisi:* modelli a regole e statistici
- **Sintassi**
  - *Teoria:* Sintassi del Linguaggio Naturale
  - *Strumenti:* CFG
  - *Analisi Sintattica:* parsing top-down, bottom-up, Early
- **Semantica**
  - Lexical Semantics
  - Sentence Semantics

Info

Sommario

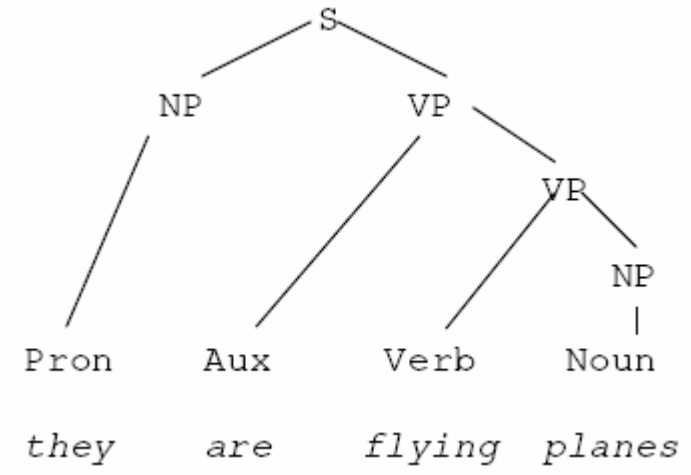
- **Strumenti per la Sintassi**

- Introduzione
- Context-Free Grammar (*CFG*)
  - Definizione
  - CFG per la sintassi
  - Limiti e problemi
- Parsing
  - Parsing a costituenti
    - Parsing Top-Down
    - Parsing Bottom-Up
    - Parsing misto (*left-corner*)
    - Chart parsing
    - Programmazione dinamica
    - Algoritmo di Earley
  - **Parsing a dipendenze**
    - **Cenni**
    - **Conversione**
  - Parser Evaluation
  - Chaos

Costituenti VS Dipendenze

Esistono due classi principali di parser, in base alla struttura prodotta in output:

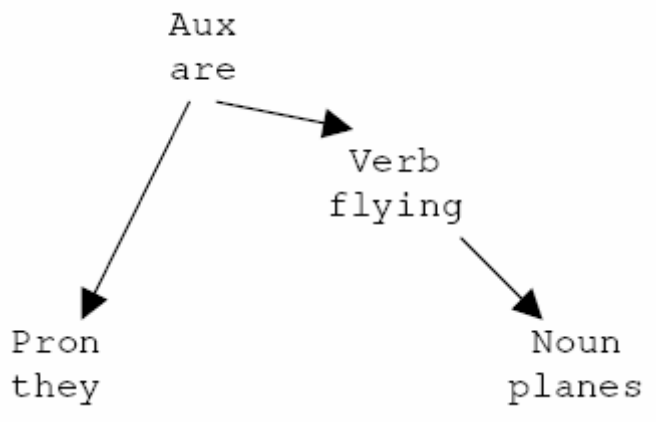
parser a costituenti



NODI: costituenti, parole

ARCHI: composizione di costituenti

parser a dipendenze



NODI: parole

ARCHI: dipendenze sintagmatiche tra una parola e un sintagma dominato da una parola

Parsing a costituenti

PARSING A COSTITUENTI

Processo di riconoscimento di una stringa e di assegnazione ad essa di una struttura sintattica corretta, ovvero un parse-tree:

- che abbia *S* come radice
- che abbia tutti e soli gli elementi della stringa come foglie

PUNTI FONDAMENTALI:

- Le CFG sono un modello dichiarativo
  - quindi non specificano come effettuare il parsing (creare l'albero)
  - al contrario, nella morfologia, gli FST sono modelli procedurali
- Le DCG possono implementare limitatamente il parsing
  - Solamente ricerca top-down depth-first
- Il parsing sintattico può essere visto come ***problema di ricerca di alberi corretti***
  - Come già avviene in morfologia con gli FST
  - Possono essere quindi usate diverse strategie

Fondamentali

Parsing a dipendenze

PARSING A DIPENDENZE

Processo di riconoscimento di una stringa e di assegnazione ad essa di una struttura sintattica corretta, ovvero un grafo alle dipendenze in cui:

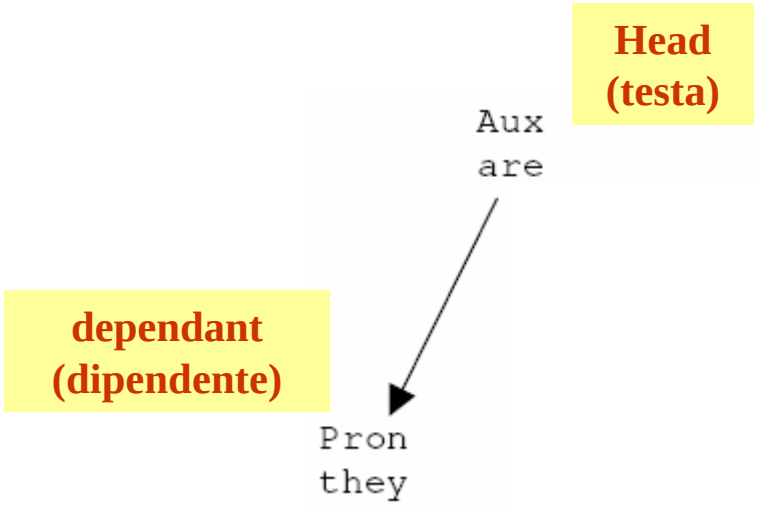
- i nodi rappresentano parolo
- gli archi (possibilmente etichettati) rappresentano le dipendenze sintagmatiche tra le parole

PUNTI FONDAMENTALI:

- I parser a dipendenze si focalizzano sulle **relazioni grammaticali** tra parole
- In ogni costituente viene evidenziato il *governatore* (**governor**) e la *testa* (**head**)
- Permettono di gestire **dipendenze lunghe**
- Esempi di modelli dichiarativi sono le Link Grammar (Sleator Temperley,1993) e le Constraint Grammar (Karlsson et al.,1995)

Parser a Dipendenze

- I nodi rappresentano le parole della frase
- Gli archi rappresentano le dipendenze grammaticali tra le parole
- Una dipendenza connette una head al suo dependant
- La head generalmente determina il comportamento (agreement) della coppia



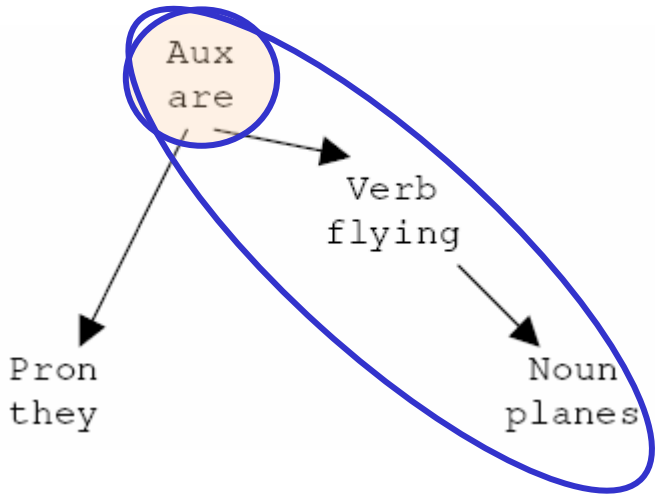
***Agreement* verbo-soggetto (unificazione)**

$$\begin{bmatrix} \text{category:noun} \\ \text{person:X} \\ \text{number:Y} \\ \text{case:nominative} \end{bmatrix} \leftarrow \begin{bmatrix} \text{category:verb} \\ \text{person:X} \\ \text{number:Y} \end{bmatrix}$$

Parser a Dipendenze

- Esempi di dipendenze:
  - verbo → oggetto
  - verbo → complemento
  - nome →modificatore
- Una struttura frasale inizia generalmente con una dipendenze in cui la head è il verbo principale della frase
- Costituenti sono costruiti connettendo ricorsivamente le dipendenze sino alla fine

Costituenti che hanno "are" come head

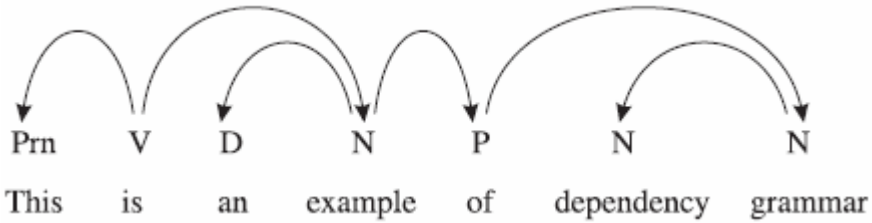


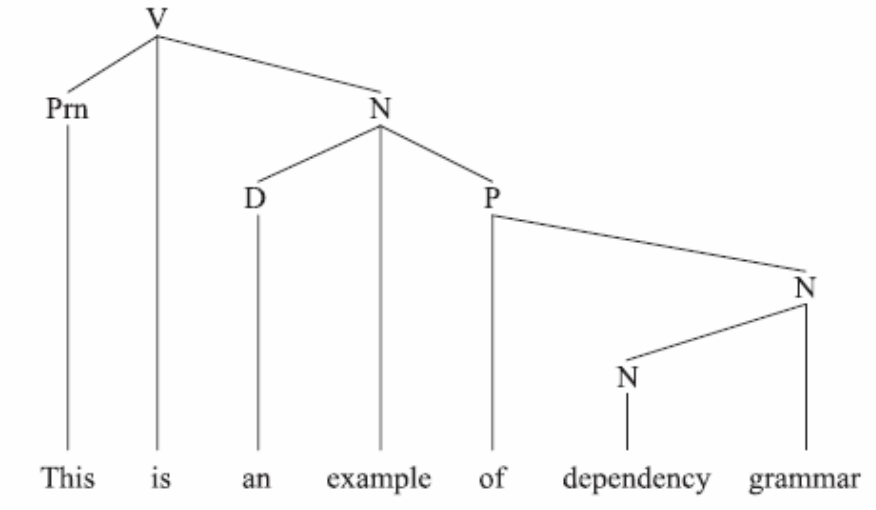
[are]

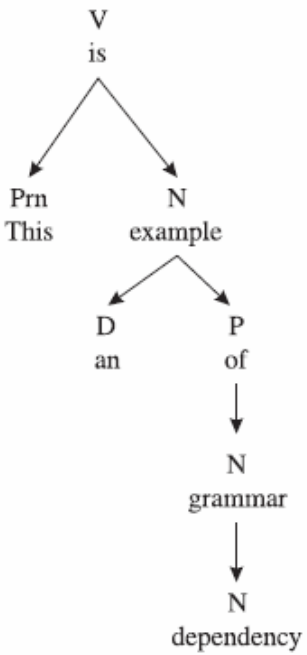
[are flying plane]

Parser a Dipendenze

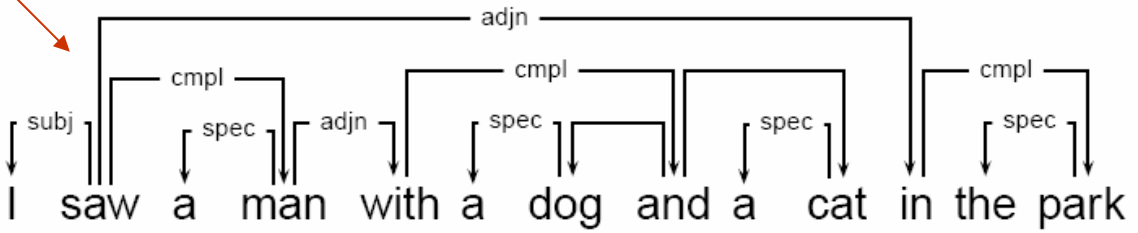
- Diversi modalità (equivalenti) di rappresentare le dipendenze:





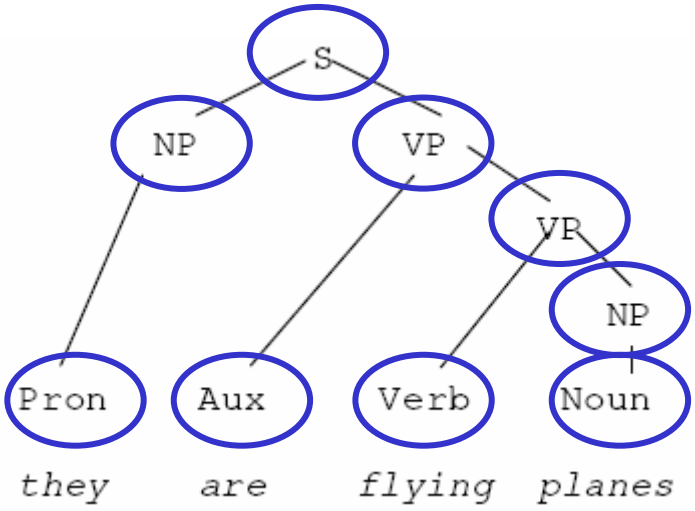


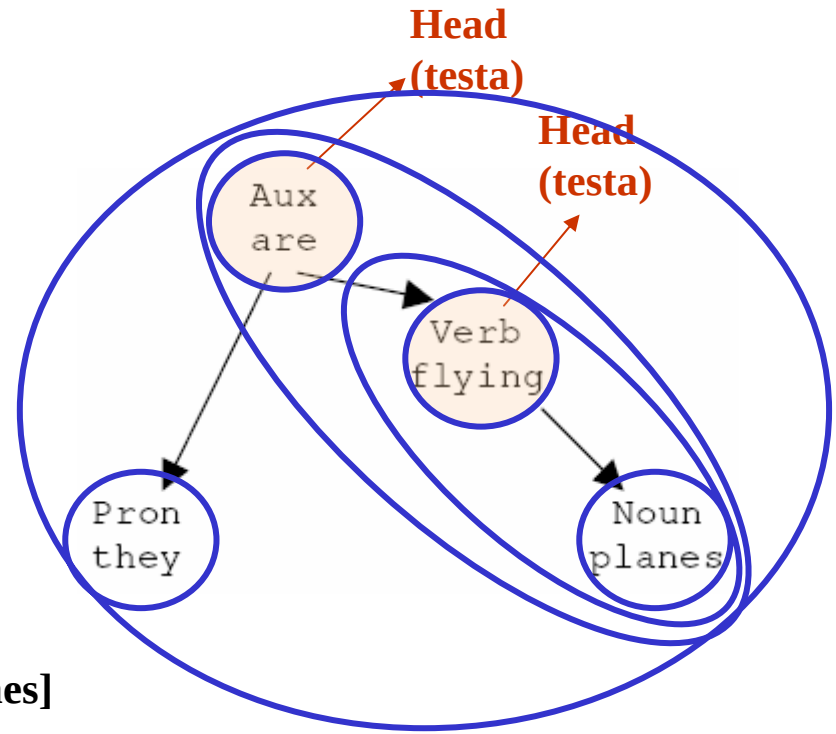
Aggiunta di etichette grammaticali sugli archi



Costituenti VS Dipendenze

- Un *parser a costituenti* modella esplicitamente i costituenti
  - Ogni nodo non terminale è un costituente
- Un *parser a dipendenze* modella implicitamente i costituenti
  - Ogni nodo è un costituente
  - Ogni sotto-albero è un costituente





Costituenti:

[they] [are] [flying] [planes]

[flying planes] [are flying plane]

[they are flying planes]

Dipendenze

Conversione Costituenti →Dipendenze

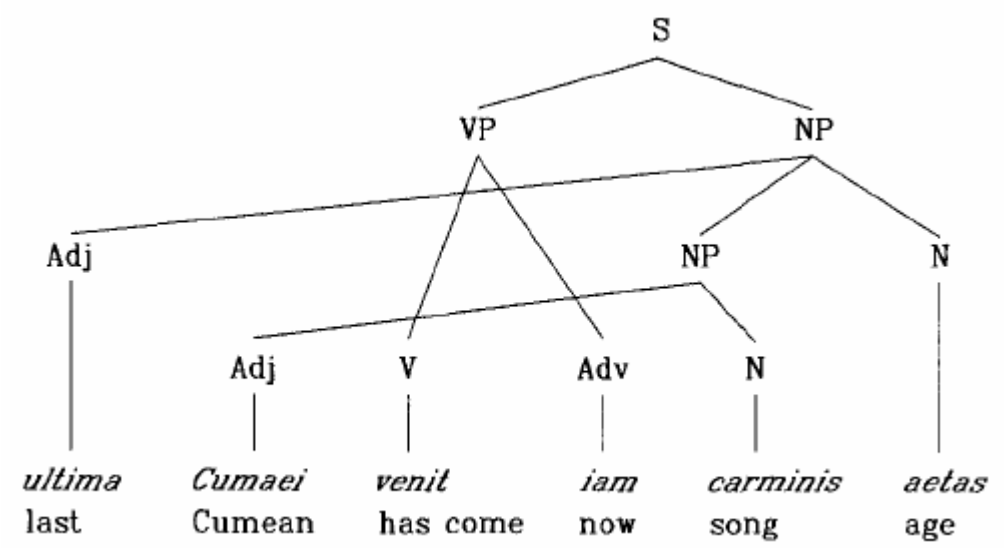
Parser a costituenti e parser a dipendenze sono ***strongly equivalent***:

- Generano / riconoscono le stesse frasi
- Producono le stesse asserzioni strutturali sulle frasi se i parser alle dipendenze sono estesi con la nozione di *head*

- Conversione dipendenze – costituenti

Due parole a favore di… Dipendenze

- Problema dei *costituenti discontinui* e delle *dipendenze lunghe*
  - In alcune lingue (es. Russo e Latino), può accadere che un costituente non sia composto da parole consecutive (costituente discontinuo)
  - Le dipendenze lunghe sono difficilmente catturabili da un parser a costituenti



Which flight do you want me to have the travel agent to book ?

Due parole a favore di… Dipendenze

- La rappresentazione a dipendenze è più vicina a formalismi semantici
- Il processo di parsing è più semplice:
  - un nodo = una parola
  - non c'è necessità di "inventare" nodi, il parser deve solo connettere parole
- La mente umana sembra operare il parsing secondo una strategia a dipendenze (Abney,1989)

Sommario

- **Strumenti per la Sintassi**

- Introduzione
- Context-Free Grammar (*CFG*)
  - Definizione
  - CFG per la sintassi
  - Limiti e problemi
- Parsing
  - Parsing a costituenti
    - Parsing Top-Down
    - Parsing Bottom-Up
    - Parsing misto (*left-corner*)
    - Chart parsing
    - Programmazione dinamica
    - Algoritmo di Earley
  - Parsing a dipendenze
    - Cenni
    - Conversione
  - **Parser Evaluation**
  - Chaos

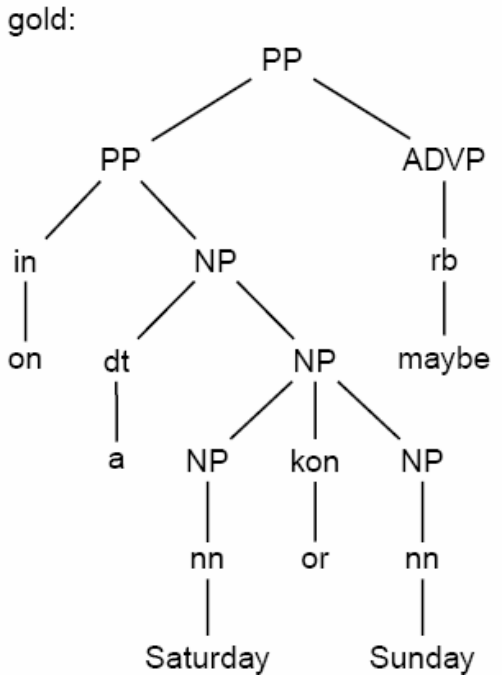
Valutazione: *parser a costituenti*

- I **parser a costituenti** sono valutati utilizzando tecniche standard (*PARSEVAL measures*)
  - si valuta la correttezza del singolo costituente
  - *costituente corretto*: costituente con la corretta produzione
  - *gold standard*: annotazione manuale della frase per valutare la correttezza

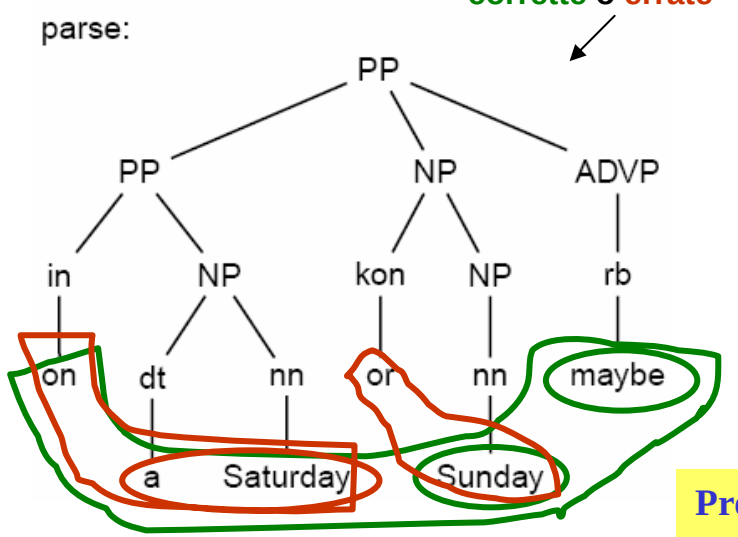
- Quattro misure:
  - **Precision** : numero di costituenti corretti prodotti dal parser, diviso il numero totale di costituenti prodotti dal parser
  - **Recall** : numero di costituenti corretti prodotti dal parser, diviso il numero di costituenti corretti nel gold standard
  - **Labeled Precision** : percentuale di costituenti corretti con la corretta etichetta prodotti dal parser
  - **Labeled Recall** : percentuale di costituenti con una data etichetta nel gold standard che sono prodotti dal parser

Valutazione: *parser a costituenti*

- Esempio (da Hinrichs-Kubler , ESSLLI-05)



Produzioni corrette o errate



Costituenti etichettati del gold standard:

[on a Saturday or Sunday maybe]	*PP*
[on a Saturday or Sunday]	*PP*
[a Saturday or Sunday]	*NP*
[Saturday or Sunday]	*NP*
[Saturday]	*NP*
[Sunday]	*NP*
[maybe]	*ADVP*

Costituenti etichettati prodotti dal parser:

[on a Saturday or Sunday maybe]	*PP*
[on a Saturday]	*PP*
[or Sunday]	*NP*
[Saturday or Sunday]	*NP*
[Sunday]	*NP*
[maybe]	*ADVP*

Precision : 3/6 = 0.5

Recall : 3/7 = 0.42

Labeled precision = 0.5

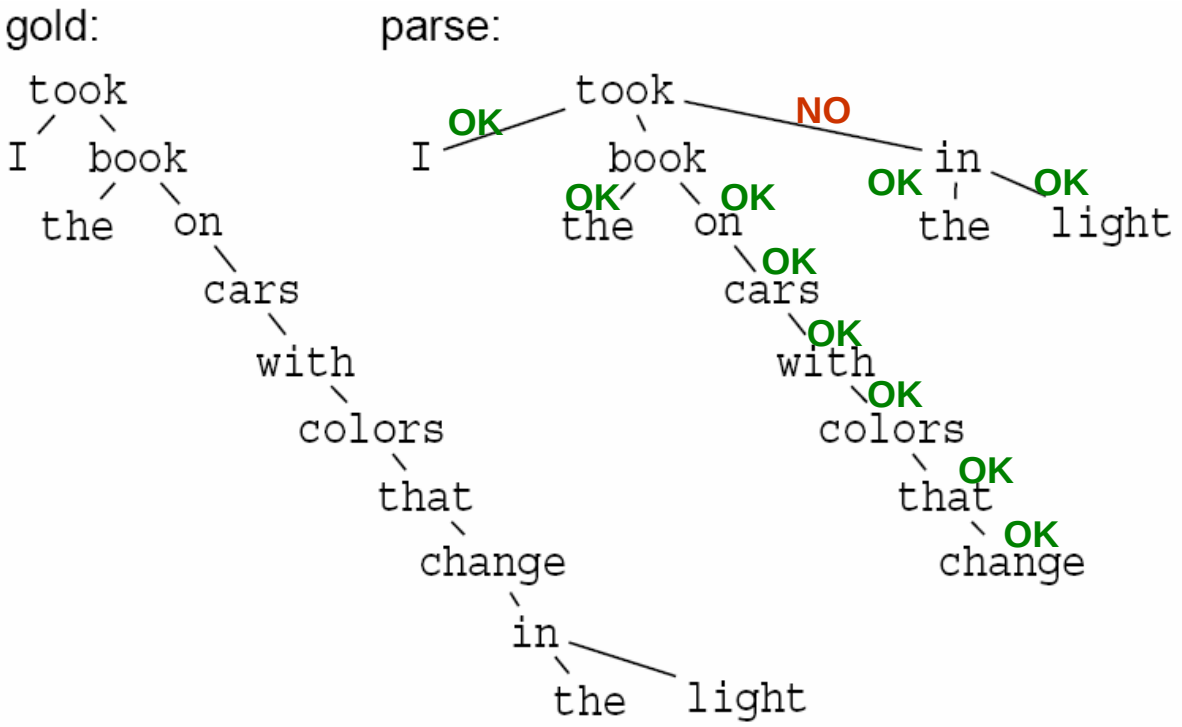
Labeled recall = 0.42

Evaluation

Valutazione: *parser a dipendenze*

- I **parser a dipendenze** sono valutati utilizzando un gold standard come riferimento. Le misure adottate sono:
  - **Precision** : numero di dipendenze corrette rispetto al gold standard
  - **Recall** : numero di dipendenze del gold standard catturate dal parser

- Esempio (da Hinrichs-Kubler , ESSLLI-05)



Precision =Recall :
11/12 = 0.92

Sommario

- **Strumenti per la Sintassi**

- Introduzione
- Context-Free Grammar (*CFG*)
  - Definizione
  - CFG per la sintassi
  - Limiti e problemi
- Parsing
  - Parsing a costituenti
    - Parsing Top-Down
    - Parsing Bottom-Up
    - Parsing misto (*left-corner*)
    - Chart parsing
    - Programmazione dinamica
    - Algoritmo di Earley
  - Parsing a dipendenze
    - Cenni
    - Conversione
  - Parser Evaluation
  - **Chaos**

Chaos: Parsing sintattico IT

Approccio ibrido: **COSTITUENTI + DIPENDENZE**

1. **Vengono identificati i chunks (particolare tipo di costituenti)**
2. **Vengono identificate le dipendenze**

Vantaggi:

- Il raggruppamento in chunks facilita l'analisi a dipendenze
- L'analisi a dipendenze consente di gestire *long distance dependecies* ed altri fenomeni
- L'utilizzo della plausibilità (certezza nell'identificare la dipendenza) permette di gestire l'ambiguità

Tre moduli fondamentali:

- Chunker : identifica i chunks
- Verb Shallow Analyzer (*VSA*) : identifica le dipendenze verbali principali
- Syntactic Shallow Analyzer (*SSA*) : identifica altre dipendenze

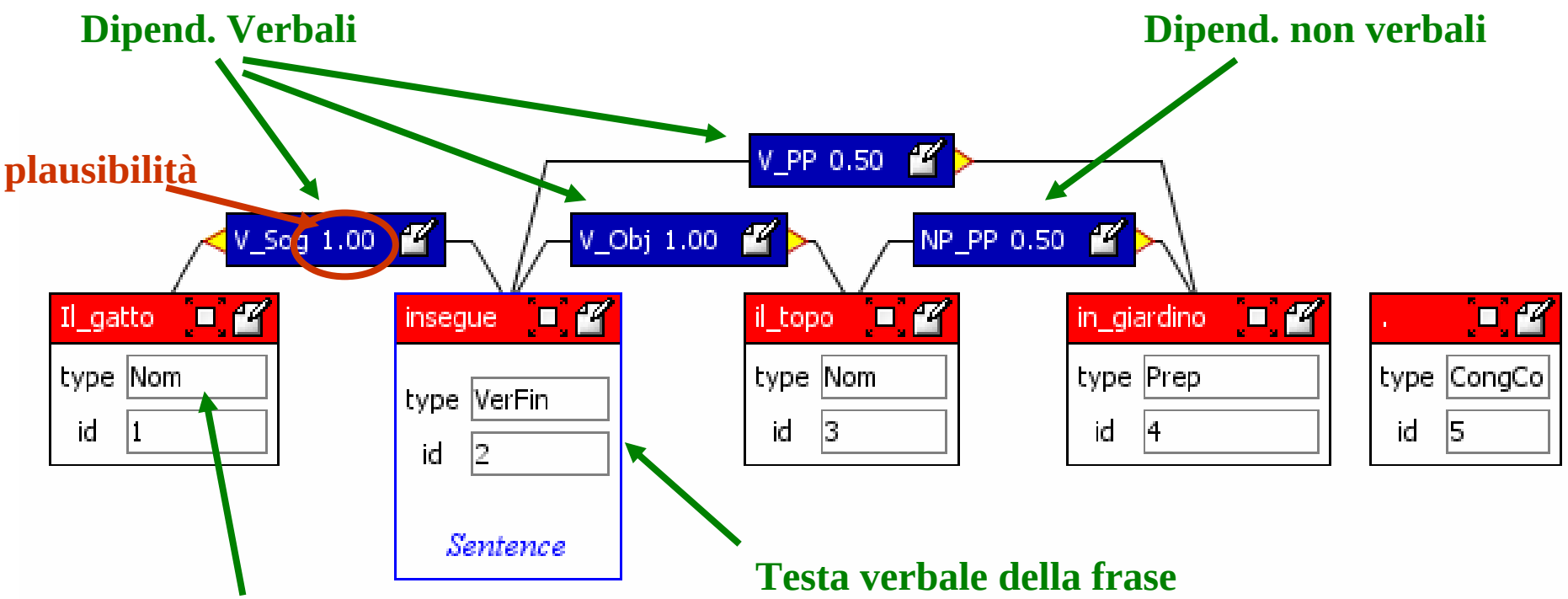
Chaos: Parsing sintattico IT

L'analisi viene rappresentata in una struttura unica:

XDG (grafo costituenti-dipendenze)

costituenti : *chunks*

Icd : *inter-chunks dependecies*



Chaos: Parsing sintattico IT

CHAOS *Chunker* `[/data/KB/it/Chunker]`

- Chunk
  - I chunk sono un particolare tipo di costituenti
  - Nucleo non ricorsivo di *sintagmi*
  - Non essendoci ricorsività, i chunk possono essere riconosciuti da FSA
  - Informazione pregressa necessaria:
    - *Morfologia*
    - *POS tagging*
- **Come funziona il Chaos Chunker:**
  - Basato su **prototipi di chunk** in *Prolog*
  - Analizza le parole da sinistra a destra, raggruppandole nel chunk più lungo possibile riconosciuto applicando le regole
  - Individua:
    - ***Governatore:*** parola che veicola il significato del chunk
    - ***Testa sintattica (head):*** parola che lega il chunk al resto della frase

Chaos: Parsing sintattico IT

CHAOS *Chunker* `[/data/KB/it/Chunker]`

- Come identificare un chunk ?
  - **Nella pratica, il *chunk* è il più grande costituente possibile che lascia la frase aperta a tutte le ammissibili interpretazioni sintattiche.**

- **ESEMPI:**
  - ***"[Il bel gatto]** mangia il topo"*

    "Il bel gatto" è chunk in quanto la sequenza DET-ADJ-NC è l'unica aggregazione possibile, ovvero l'ADJ è legato sempre come specificazione dell'NC in tutte le frasi italiane
  - "***[Il gatto] [del vicino]*** magia il topo"

    "Il gatto del vicino" è un costituente, ma non è un chunk, in quanto la sequenza DET-NC-PREP-NC non è l'unica interpretazione sintattica possibile. Ovvero esistono sequenze di questo tipo in cui PREP-NC non sono specificazioni di NC, ma di un altro elemento della frase, ad esempio: *"Il gatto nel prato mangia", "nel prato"* specifica *"mangia"*
  - "***[Il gatto] [interessato]*** al topo"

    "Il gatto interessato" non è un chunk, in quanto la sequenza DET-NC-ADJ non garantisce tutte le interpretazioni sintattiche possibili. Ovvero, il *PREP-NC* seguente in alcuni casi specifica DET-NC (es.*"Il gatto interessato al topo"*), in altri ADJ (*"Il gatto interessato del vicino"*)

Chaos: Parsing sintattico IT

CHAOS *Chunker* [/data/KB/it/Chunker]

- Prototipi Prolog per il riconoscimento di semplici chunk aggettivali:

```
constituent_class([_cst1], 'Agg', _mor, 1, 1):-
        adjective(_cst1),
        common_morfology(_cst1, _mor).
```

ADJ

```
constituent_class([_cst1, _cst2], 'Agg', _mor, 1, 2):-
        adverb(_cst1),
        adjective(_cst2),
        common_morfology(_cst1, _cst2, _mor).
```

ADV
ADJ

```
constituent_class([_cst1, _cst2, _cst3], 'Agg', _mor, 1, 1):-
        adjective(_cst1),
        coordinative_conjunction(_cst2),
        \+(comma(_cst2)),
        adjective(_cst3),
        common_morfology(_cst1, _cst3, _mor).
```

ADJ
COORD
ADJ

Chaos: Parsing sintattico IT

CHAOS *Chunker* [/data/KB/it/Chunker]

COMPLEX CONSTITUENT
(chunk)

head

SIMPLE CONSTITUENT
(sotto-costituenti del chunk)

governor

Chaos: Parsing sintattico IT

CHAOS *ChunkTypes in* IT

Agg	Chunk aggettivali
Avv	Chunk avverbiale
CongCo	Chunk coordinativo
CongSub	Chunk subordinativo
Nom	Chunk nominale
Prep	Chunk preposizionale
VerFin	Chunk verbale finito
VerGer	Chunk verbale gerundivo
VerInf	Chunk verbale infinito
VerPart	Chunk verbale participio
VerPred	Chunk verbale predicativo aggettivale
VerNom	Chunk verbale nominale
VerPrep	Chunk verbale preposizionale
?	Chunk sconosciuti

Chaos: Parsing sintattico IT

CHAOS *VSA* `[/data/KB/it/SubcatLexicon]`

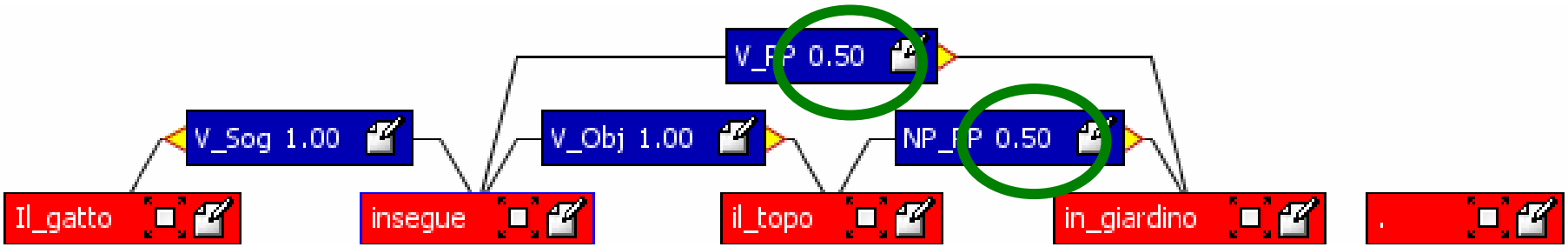
- Identifica le dipendeze (*icd*) verbali all'interno della struttura
  - Utilizza un lessico di patterns di sottocategorizzazione verbale (*LIFUV.lex*)
  - 1844 sottocategorizzazioni
  - Codificate manualmente ed apprese automaticamente da corpora
  - Una dipendenza ricavata dal lessico ha **plausibilità=1**

```
pattern(accrescere,[[(oggetto,Post)]]).
pattern(accucciare,[[]]).
pattern(accumulare,[[(su,Post)],[(oggetto,Post)],[(oggetto,Post)]]).
pattern(acquisire,[[(oggetto,Post),(a,Post)],[(oggetto,Post)]]).
pattern(acquistare,[[(in,Post)],[(oggetto,Post),(da,Post)]]).
pattern(adagiare,[[(su,Post)],[(oggetto,Post),(su,Post)]]).
pattern(addolcire,[[(oggetto,Post)],[(oggetto,Post)]]).
```

Chaos: Parsing sintattico IT

CHAOS *SSA* [/data/KB/it/SSG]

- Identifica le dipendeze (*icd*) che non sono state identificate dall'SSA:
  - *Modificatori verbali*: espressioni temporali / spaziali …
  - *Modificatori nominali*: sintagmi preposizionali / specificatori aggettivali …
- **Come funziona:**
  - E' un parser specifico basato su una grammatica discontinua
  - Consente di trovare dipendenze tra chunk non adiacenti (*gap*) → anche *long distance dependencies*
  - **<u>Ambiguità:</u>** più dipendenze possono essere prodotte per uno stesso chunk
  - Viene utilizzata la **pausibilità** per modellare l'ambiguità



Chaos: Parsing sintattico IT

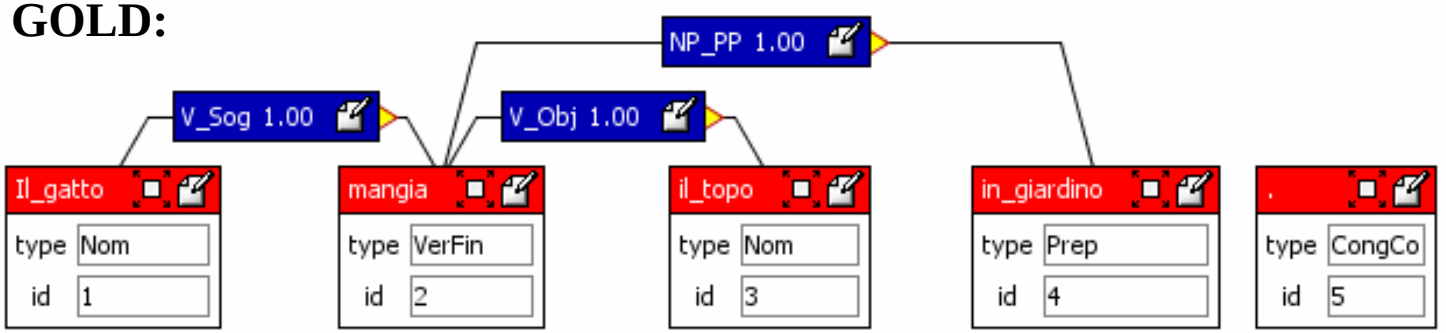
CHAOS *ICDs Types* *(VSA+SSA)*

Grammatical Subject	V_Sog
Grammatical Object	V_Obj
Indirect Object	V_NP
Verb Preposition Modifier	V_PP
Verb Adverb Modifier	V_Adv
gruppo Nominale Nominale	NP_NP
gruppo Nominale Preposizionale	NP_PP
gruppo Preposizionale Preposizionale	PP_PP
gruppo Nominale Aggettivo	NP_Adj
gruppo Nominale Participio	NP_VPart
gruppo Preposizionale Aggettivo	PP_Adj
gruppo Preposizionale Participio	PP_VPart
gruppo Aggettivo Preposizionale	Adj_PP
gruppo Avverbio Preposizionale	Adv_PP
Congiunzione coordinativa tra costituenti	x_Cong_x
gruppo Costituente Congiunzione	x_Cong
gruppo Costituente Subordinata	x_Sub
gruppo Verbo Congiunzione Subordinativa	V_CSub

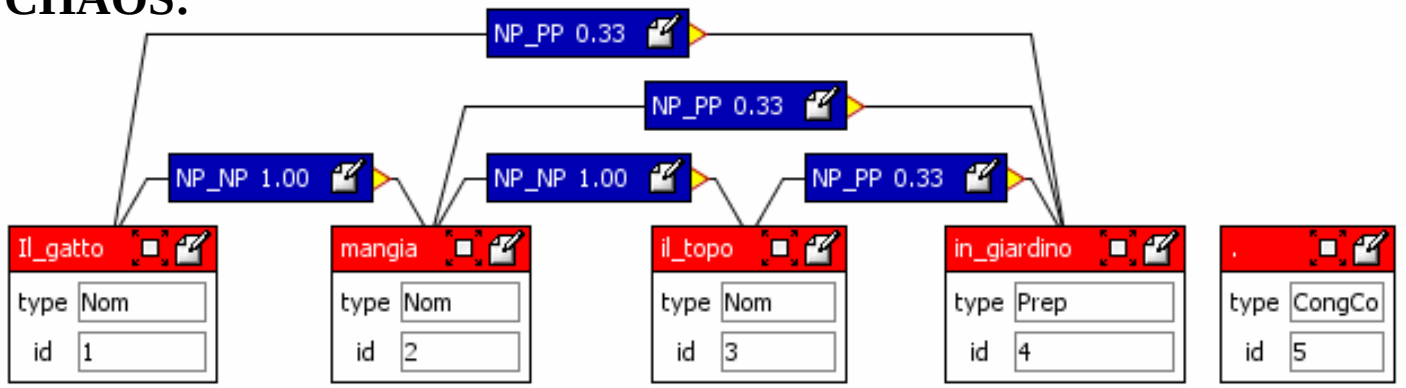
Chaos: Valutazione

- Utilizzare :
  - Le misure per i parser a costituenti per valutare i chunk
  - Le misure per i parser a dipendenze per valutare gli ICD
  - **Non** considerare le plausibilità nella valutazione
- **Esempio:**

GOLD:



CHAOS:



Chunks:

Prec = 5/5 = 1

Rec = 5/5 = 1

Lab Prec = 4/5= 0.8

Lab Rec = 4/5= 0.8

ICD:

Prec = 1/5 = 0.2

Rec = 1/3 =0.3

Progetto : Note

- **ATTENZIONE: è necessario aggiornare i tipi di ICD di Chaos prima di effettuare la seconda parte del progetto!!**

  - Scaricare il file **`icdTypes.xml`** dal sito web del progetto:

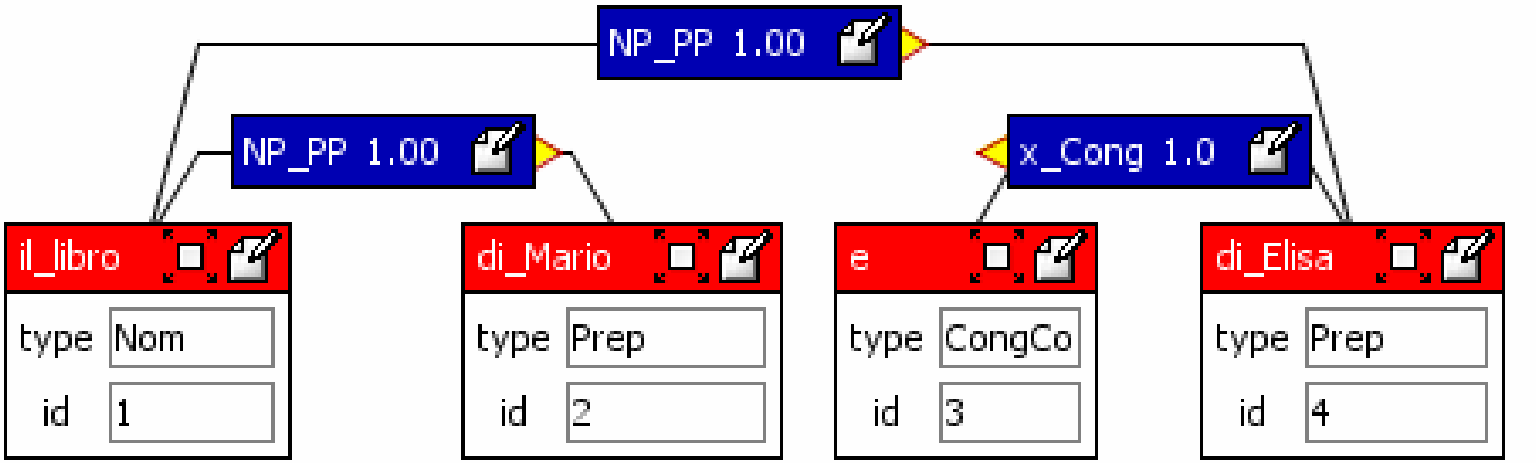
    http://ai-nlp.info.uniroma2.it/pennacchiotti/teaching/icdTypes.xml

  - Sovrascrivere il vecchio file `icdTypes.xml` in **`chaos.jar`** con quello scaricato
    - `Chaos.jar` si trova in `/lib`
    - `icdTypes.xml` ha nel JAR il seguente percorso:

      `chaos/alternatives/data/it`

Progetto : Note

- **Costrutti frasali complessi in Chaos:**
  - **Congiunzione**



Progetto : Note

- **Costrutti frasali complessi in Chaos:**
  - **Subordinazione**