CISCL
Cristiano Chesi (chesi@media.unisi.it)

NOZIONI FORMALI DI RICORSIVITÀ

Grammatiche formali

(1) **A** = **Alfabeto**: insieme finito di caratteri (A* = l'insieme di tutte le stringhe possibili costruite concatenando elementi di A; ε è l'elemento nullo)
V = **Vocabolario**: insieme (potenzialmente in)finito di parole, costruite concatenando elementi di A ($V \subseteq A^*$)
L = **Linguaggio**: insieme (potenzialmente in)finito di frasi, costruite concatenando elementi di V ($L \subseteq V^*$)

(2) Una **grammatica formale** per il linguaggio L, $\mathbf{G_L}$, è un insieme di regole che permettono di generare/riconoscere tutte e sole le frasi appartenenti a L (capacità generativa debole) e di assegnare a queste frasi un'adeguata descrizione strutturale (capacità generativa forte). G_L deve essere:
esplicita (il giudizio di grammaticalità deve essere frutto solo dell'applicazione meccanica delle regole scelte)
consistente (una stessa frase non può risultare allo stesso tempo grammaticale e non grammaticale)

(3) Una **grammatica a struttura sintagmatica** (Chomsky 1957) può essere definita come una quadrupla
<V, $\mathbf{V_T}$, →, **S>** tale che:

V vocabolario della lingua

$\mathbf{V_T}$ vocabolario terminale ($V_T \subseteq V$; V_N è il complementare di V_T rispetto a V, cioè il voc. non terminale)

→ insieme di regole di riscrittura in genere nella forma $\varphi A \psi \rightarrow \varphi \tau \psi$ ($A \in V_N$; $\varphi, \tau, \psi \in V_T$)

S insieme degli assiomi (solitamente simbolo iniziale $S \in V_N$)

(4) Una **derivazione** è una sequenza finita di regole di riscrittura che parte dall'assioma (S) e arriva fino a generare una stringa composta solo da V_T. Ogni passo della derivazione è una stringa composta dalla concatenazione di elementi di V.

(5) Un **albero** viene creato, seguendo la derivazione, ramificando un elemento ogni volta che questo viene riscritto seguendo una regola. Minimalmente una **descrizione strutturale** è una tripla **<I**, **P**, **D>** tale che:

I è un insieme finito di identificatori (i.e. elementi lessicali, inclusiveness condition, Chomsky 1995)

P è un insieme finito di relazioni di precedenza immediata (diverso dal classico ordine totale stretto, questa relazione è definita solo tra elementi adiacenti: ad es. <A, B> significa A *precede immediatamente* B;
P = {<A, B>, <B, C>, <C, D>}

D è un insieme finito di relazioni di dominanza immediata (diverso dalla classica relazione di dominanza, questa è una relazione *parziale*, *binaria*, *transitiva* e *asimmetrica*: ad es. A < B, significa A *domina immediatamente* B;
D = {A < B, B < C, C < D} (equivalente a $[_A A [_B B [_C C D]]$)

(6) La **gerarchia di Chomsky** (1956, 59) pone in relazione grammatiche di potenza diversa ponendo restrizioni sulla struttura delle regole:

Tipo 0: grammatiche non ristrette (**Turing equivalent**):	$\alpha \rightarrow \beta$	$(\alpha \neq \varepsilon)$	[ATNs]
Tipo 1: grammatiche contestuali (**context-sensitive**):	$\alpha A \beta \rightarrow \alpha \gamma \beta$	$(\gamma \neq \varepsilon)$	[TAGs]
Tipo 2: grammatiche non-contestuali (**context-free**):	$A \rightarrow \gamma$		[PSGs]
Tipo 3: grammatiche regolari:	$A \rightarrow xB$		[FSA]

(7) Esistono metodi effettivi per determinare le proprietà che non possono essere catturate da una classe di grammatiche (**pumping lemmas**)

(8) Le lingue naturali **non sono generabili da grammatiche regolari** (Chomsky 1956):
If A then B (con A e B potenzialmente anch'esse nella forma "if X then Y"... quindi linguaggi di tipo $a^n b^n$)

(9) Le lingue naturali **non sono generabili da grammatiche context-free** (Shieber 1985):
"famoso" dialetto svizzero tedesco ("ABC...ABC"... quindi linguaggi di tipo XX)

(10) Le **grammatiche context-sensitive** però sono "**troppo costose**" in termini computazionali. Si cercano quindi grammatiche "**Mildly-Context-Sensitive**"

(11) **Minimalist Grammars** (Stabler 1997) {**V**, **Cat**, **Lex**, **F**}:

V is a set of non-syntactic features, (P ∪ I) where *P* are phonetic features and *I* interpretable (semantic) features;

Cat is a set of syntactic features, *Cat* = (*base* ∪ *select* ∪ *licensors* ∪ *licensees*) where

base are standard categories {*complementizer*, *tense*, *verb*, *noun* ...},

select specify one of the three postulated kind of selection {=*x*, =*X*, *X*= | x ∈ *base*};

licensees specify requirement asking for phrasal movement to be satisfied {-*wh*, -case ...}, -x triggers covert movement, while -X would trigger overt movement;

licensors are the features able to satisfy licensee requirements {+*wh*, +case ...}

Lex is a finite set of expressions built from *V* and *Cat* (the lexicon);

F is a set of the two partial functions from tuples of expressions to expressions {*merge*, *move*};

Esempio di MG:

V = *P* {/what/, /did/, /you/, /see/}, *I* {[what], [did], [you], [see]}

Cat = *base* = {D, N, V, T, C}, *select* = {=D, =N, =V, =T, =C}, *licensors* {+wh}, *licensees* {-wh}

Lex = {-wh D what }, {=V T did}, {D you}, {=D D= V see}, {=T +wh C ∅}

F = {*merge*, *move*} such that:

merge (X, Y) = is a function taking two subtrees X and Y, outputting an unified structure Z if and only if X has a selecting feature (=*f*, =*F*, *F*=) and Y has the needed selected feature *F* or vice versa

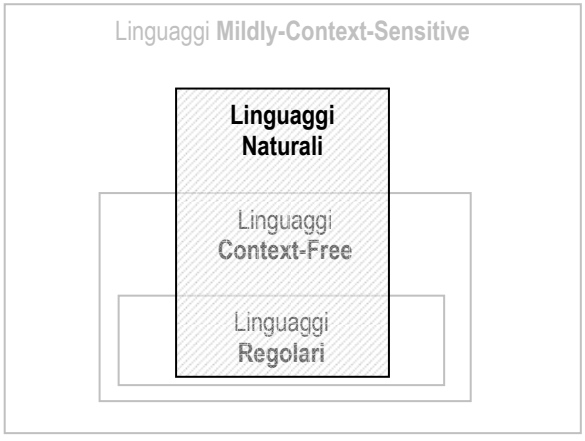
move (X, Y) = is a function taking two subtrees [+*g* X] and [-*g* Y] such that <[+*g* X], W, [-*g* Y]> (with W, that can be any possible subtree, even a null one, but without any selecting/selector feature *g*) and produces Z of the form [Y, X, W, t_Y]

Esempio di derivazione:

1. merge ([=D D= V see], [-wh D what]) → [$_V$ D= V see, -wh what]
2. merge ([D you], [D= V see, -wh what]) → [$_V$ you, [$_V$ V see, -wh what]]
3. merge ([=V T did], [$_V$ you, [$_V$ -t V see, -wh what]]) → ([$_T$ +t T did, [$_V$ you, [$_V$ -t see, -wh what]]]
4. merge ([=T +wh C ∅], [$_T$ T did, [$_V$ you, [$_V$ see, -wh what]]]) → ([$_C$ +wh C ∅, [$_T$ did, [$_V$ you, [$_V$ see, -wh what]]]])
5. move ([$_C$ +wh C ∅, [$_T$ did, [$_V$ you, [$_V$ see, -wh what]]]] → [$_C$ What C ∅, [$_T$ did, [$_V$ you, [$_V$ see, t_{what}]]]]

(12) a. gerarchia attuale

b. reali risultati di inclusione attesi



CISCL
Cristiano Chesi (chesi@media.unisi.it)

Tipi di ricorsività

(13) Una regola si dice ricorsiva se un simbolo *X* viene rimpiazzato da una stringa di simboli che contiene *X* stesso:

$X \rightarrow aXb$ (con *X* non terminale e *a* e *b* terminali, non terminali o nulli)

Si incorre in ricorsione anche quando questa situazione occorre dopo l'applicazione di un insieme di regole individualmente non ricorsive:
$X \rightarrow aYb$ $Y \rightarrow bX$

(14) Si dice **iterazione** una regola come quella in (13) con *a* oppure *b* nulli:

$X \rightarrow aX$ *(linguaggi ricorsivi a destra)*

$X \rightarrow Xb$ *(linguaggi ricorsivi a sinistra)*

i linguaggi **ricorsivi a destra** o a **sinistra** sono generabili da **grammatiche regolari**

(15) Si dice **counting recursion** una regola come quella in (13) con *a* oppure *b* entrambi NON nulli:

$X \rightarrow aXb$

$X \rightarrow \varepsilon$

(*ab*, *aabb*, *aaabbb*, *aaaabbbb* ... il nome deriva dal fatto che per riconoscere/generare una stringa appartenente a questa grammatica si devono contare le occorrenze di *a* e *b*)
le **counting recursions** (illimitate) sono generabili da **grammatiche context-free**

(16) Si dice **mirror recursion** una regola come quella in (13) con *a* oppure *b* entrambi NON nulli e "simmetrici" rispetto a *X*:

$X \rightarrow aXa$

$X \rightarrow bXb$

$X \rightarrow \varepsilon$

le **mirror recursion** sono anch'esse generabili da **grammatiche context-free**

(17) Si dice **identity recursion** una regola leggermente diversa dalle precedenti che può venire (impropriamente) rappresentata come segue:
$X \rightarrow YY$

le **identity recursion** NON sono generabili da **grammatiche context-free**

Ricorsività linguistica e limitazioni

(18) a. **Incassamento a destra** (ab^nc**: iterazione**):
[il cane morse [il gatto [che rincorse [il topo [che scappò]]]]]

b. **Incassamento centrale** (a^nb^n**: counting recursion**):
[il cane [che il gatto [che il topo che scappò], rincorse], morse]

c. **Dipendenze cross-seriali** (xx, **identity recursion**)
Gianni, Maria e Marco sono rispettivamente sposato, nubile e divorziato

(19) Fitch & Hauser (2004) mostrano come i primati non umani riescano a cogliere violazioni solo nei linguaggi regolari $((ab)^n)$ e non nelle grammatiche a struttura sintagmatica (a^nb^n) (ma è veramente una grammatica a struttura sintagmatica quella chiamata in causa? Kochacski 2005)

(20) Christiansen & Charter (1999) mostrano come le reti ricorsive (SRN) riescano a catturare meglio le dipendenze cross-seriali dell'incassamento centrale (dati peraltro psicolinguisticamente plausibili, Bach & al. 1986... ma le SRN cosa apprendono veramente?)

Cosa produce ricorsività in una grammatica minimalista?

(21)
Lexicon: insieme di tratti assemblati in parole (ipotesi lessicalista: le flessioni, ad es., sono elaborate nel lessico)
Select: selezione di un insieme di tratti assemblati (Lexical Items: LI)
Numeration: un insieme di coppie (LI, index) (Chomsky 1995, poi dopo l'index non viene più considerato)
Derivazione: una serie di applicazioni di **merge** e **move**
Phase: PH = [α[H β]] ; "oggetto sintattico" derivato dalla scelta di un LA (Lexical Array = insieme di LI) e dall'applicazione, fino ad esaurimento LA, di merge & move (Chomsky 2000: 106), impenetrabile, ad eccezione dell'edge α-H, (PIC) dopo lo Spell-Out viene applicato a PH (solo β viene "prodotto"; nel frattempo ogni operazione applicabile è stata applicata ad LA).

(22) Operazioni di **costruzione della struttura**:

Merge (Chomsky 1995: 396)
The simplest object constructed from α and β is the set {α, β}, so we take γ to be at least this set.

External merge
α and β are separate objects (costruisce la struttura argomentale)

Set merge (Chomsky 2001: 6, 18)
Takes two elements α, β already constructed and creates a new one consisting of the two; in the simplest case {α, β}.

Pair merge (Chomsky 2001: 18).
"But it is an empirical fact that there is also an asymmetric operation of adjunction, which takes two objects β and α and forms the ordered pair <α, β >, α adjoined to β. Set-merge and pair-merge are descendants of *substitution* and *adjunction* in earlier theories. Given the basic properties of adjunction, we might intuitively think of a as attached to β on a separate plane, with β retaining all its properties on the "primary plane," the simple structure."

"Richness of expressive power requires an operation of predicate composition: that is not provided by set-Merge.... But it is the essential semantic contribution of pair merge". (Chomsky 2001: 18)

Internal merge (i.e. **move**) (Chomsky 2001: 9)
If one is part of the other, e.g. if β is part of α, in which case β is said to be a "copy" of its occurrence in α. (costruisce "scopal and discourse-related properties")
Giustificato dalla necessità di cancellare tratti non-interpretabili

Agree
Un tratto di un Probe (Attractor) individua un tratto di un Goal e riesce a soddisfarlo (potenzialmente anche a distanza, altrimenti si ricorre a move)

(23) Nozioni di **economia**:
Featural ciclicity (Richards 1997): cancella i tratti appena puoi!
Last resort (Chomsky 1998): merge < move
Minimal Link Condition: muovi il più vicino possibile
Greed: fai quello che devi fare il prima possibile
Procrastinate: ricorri al movimento solo come ultima risorsa

(24) **Restrizioni "empiriche":** non tutte le opzioni possibili sono grammaticali:

a. * re-merge [A [$_B$ B t_A]]
b. * self-merge [A [$_A$ B t_A]]
c. * move & project [$_A$ A [... [B t_A]]]
d. * lowering [t_A [...[B A]]]
e. * movement from nothing [A [... [$\varnothing$ t_A]]]
f. $^{\surd}$orphans formation [A [... [$_A$ t_A B]]] (head movement)
g. $^{\surd}$movement from "specifier" [A [... [$_B$ t_A B]]]

Da cosa può essere limitata la ricorsività

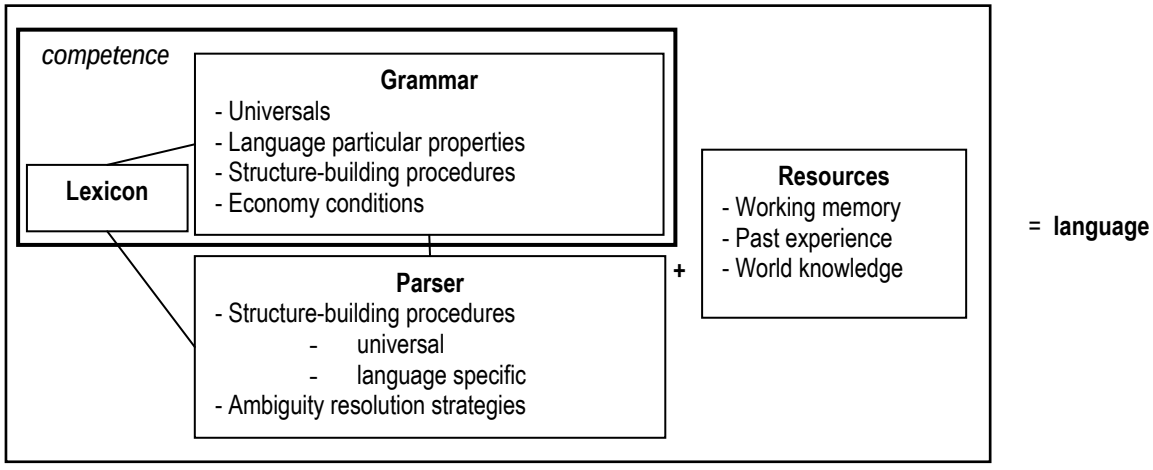
(25) Derivazioni Vs. rappresentazioni:

theories for movement	**representational**	**derivational**

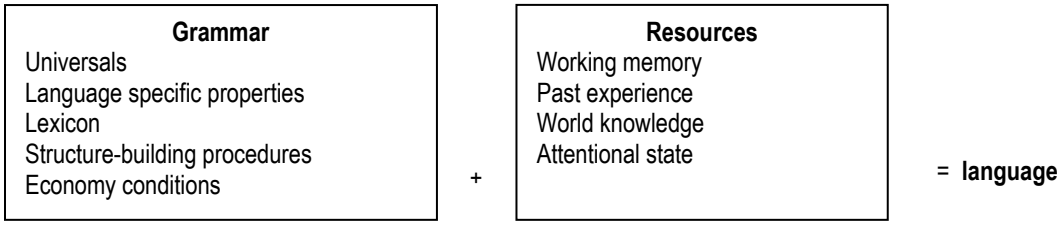
i. completeness of SDs	*unique* and *complete*: all instances present in the sentence appear in the chain:
	
	<x, tx_n, tx_{n-1} ... tx_0>
		partial: only the relevant element (a segment, at best) of the "chain" is accessed at any step:
		
		step 1: x;
		step 2: tx_n (x = tx_n);
		...
		step n: tx_0 (tx_0 = tx_1);

ii. principle/rules ordering	*irrelevant*: any order would postulate the same traces and discard ungrammatical options	*strictly defined*: unless we define extra backtracking algorithms, postulating a wrong movement would prevent the derivation from retrieve the correct SD
iii. relation among elements	*absolute*: any relational property among elements in the chain is valid within a single SD	*relative*: the relational property are valid only within a relevant lapse of time τ_n (at τ_n: <tx_n, tx_{n-1}>), then further operation would not have access anymore to the single constituents that established relations at τ_{n-1}
iv. nature of constraints	*filters* on the unique resulting representation (such as *case filter*)	*constraints* on operation application (such as *shortest move*)
v. processing implications	*none*: any order of principle/filter application would lead to the very same chain	*rigid order* of operation application: applying in advance an operation that should be applied then, would prevent the derivation from applying intermediate operations that would lead to the correct SD

(26) Modello standard:



(27) Modello PIG:



References

Bach, E., Brown, C., & Marslen-Wilson, W. (1986). Crossed and nested dependencies in German and Dutch: A psycholinguistic study. *Language and Cognitive Processes, 1,* 249–262.

Chomsky, N. (1956) Three models for the description of language. *IRE transactions on information theory,* IT-2**,** 113-124.

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

Chomsky, N. (1959) On certain formal properties of grammars. *Information & Control***,** 91-112.

Chomsky, N. (1995) *The minimalist program,* Cambridge, Mass., The MIT Press.

Chomsky, N. (1999) *Derivation by phase,* Cambridge, MA, MIT Working Papers in Linguistics.

Chomsky, N. (2001) *Beyond explanatory adequacy,* Cambridge, MA, MITWPL.

Christiansen, M. H, Chater N. (1999) *Toward a connectionist model of recursion in human linguistic performance*. Cognitive Science, 1999.

Miller, G. A., Chomsky, N. (1963) Finitary models of language users. IN D. R. LUCE, R. R. B., AND E. GALANTER (Ed.) Handbook of Mathematical Psychology. New York, John Wiley.

Shieber, S. (1985) Evidence against the context-freeness of natural language. *Linguistics and Philosophy,* 8**,** 333-343.

Stabler, E. (1997) *Derivational minimalism*. in Retoré, ed. Logical Aspects of Computational Linguistics. Springer, pp.68-95