

Unità Didattica 2

I Linguaggi di Programmazione

Linguaggio naturale e linguaggio macchina

- La comunicazione uomo-macchina avviene attraverso formalismi che assumono la forma di un linguaggio.
- Caratteristiche del **Linguaggio Naturale**:
 - Vantaggi:
 - Ricchezza espressiva;
 - Svantaggi:
 - Ambiguità;
 - Ridondanza.
- Caratteristiche del **Linguaggio Macchina** (codice binario):
 - Vantaggi:
 - Legato alla struttura fisica dell'elaboratore;
 - Potente e veloce;
 - Svantaggi:
 - Programmi lunghi e di difficile scrittura;
 - Difficoltà di messa a punto dei programmi.

Linguaggio di Programmazione

- Programma:
 - formulazione di un algoritmo nei termini di un linguaggio di programmazione.
- Linguaggio di Programmazione:
 - Linguaggio intermedio fra il linguaggio macchina e il linguaggio naturale;
 - Descrive gli algoritmi con una ricchezza espressiva comparabile con quella dei linguaggi naturali;
 - Descrive gli algoritmi in modo rigoroso.



Linguaggio Naturale

Linguaggio di Programmazione

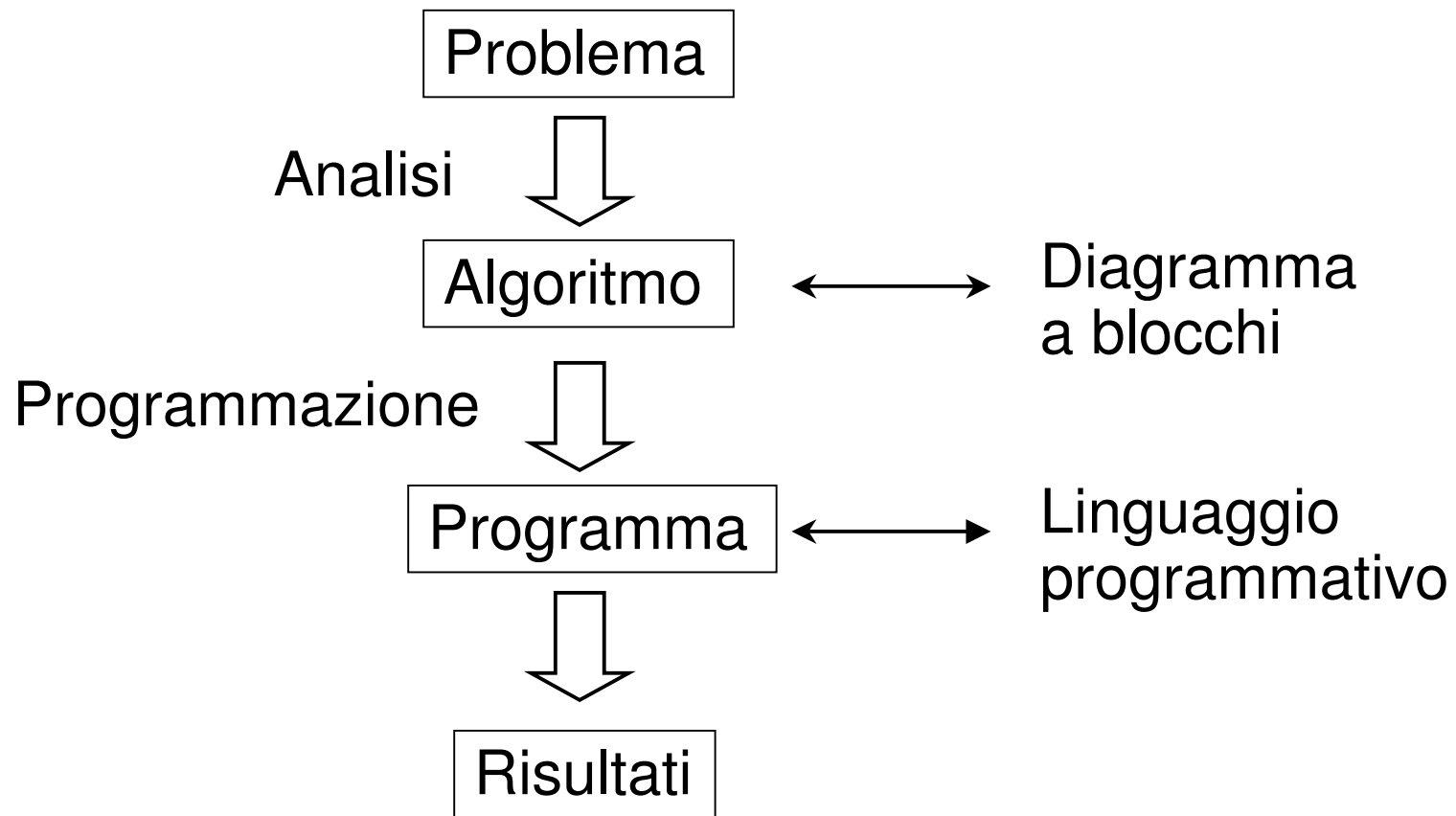
Linguaggio Macchina

Grammatica

Traduttore



Uso dei Linguaggi di Programmazione

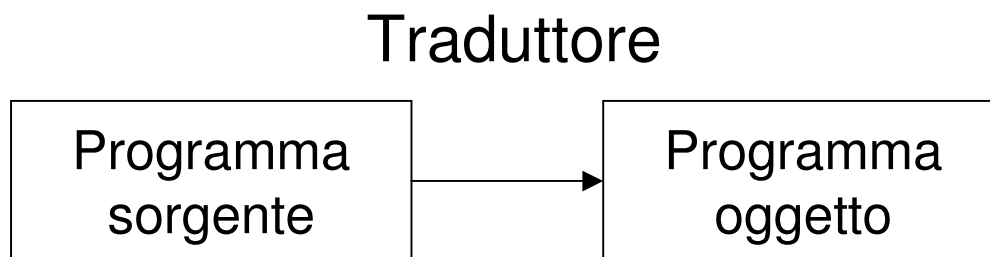


Traduttori

- Il linguaggio macchina è l'unico linguaggio compreso dall'elaboratore;
- Qualsiasi altro linguaggio di programmazione ha bisogno di un traduttore (non è possibile progettare un traduttore per i linguaggi naturali);
- I linguaggi di programmazione sono comprensibili sia dalla macchina (attraverso un traduttore) che dall'uomo.

Traduttori

- Programma sorgente:
 - Programma espresso in linguaggio di programmazione;
- Programma oggetto:
 - Programma espresso in linguaggio macchina;
- Traduttore:
 - Programma scritto in linguaggio macchina che traduce un programma sorgente in un programma oggetto;
 - E' funzione del linguaggio di programmazione e dell'architettura dell'elaboratore;



Programma sorgente
e programma oggetto
sono equivalenti (stessi
risultati sugli stessi dati)

Tipi di Traduttori: Assemblatori, Compilatori e Interpreti

Linguaggi ad alto
livello (orientati all'uomo)

Linguaggi a basso
livello (linguaggi assemblativi,
orientati alla macchina)

Compilatori
o Interpreti

Assemblatori

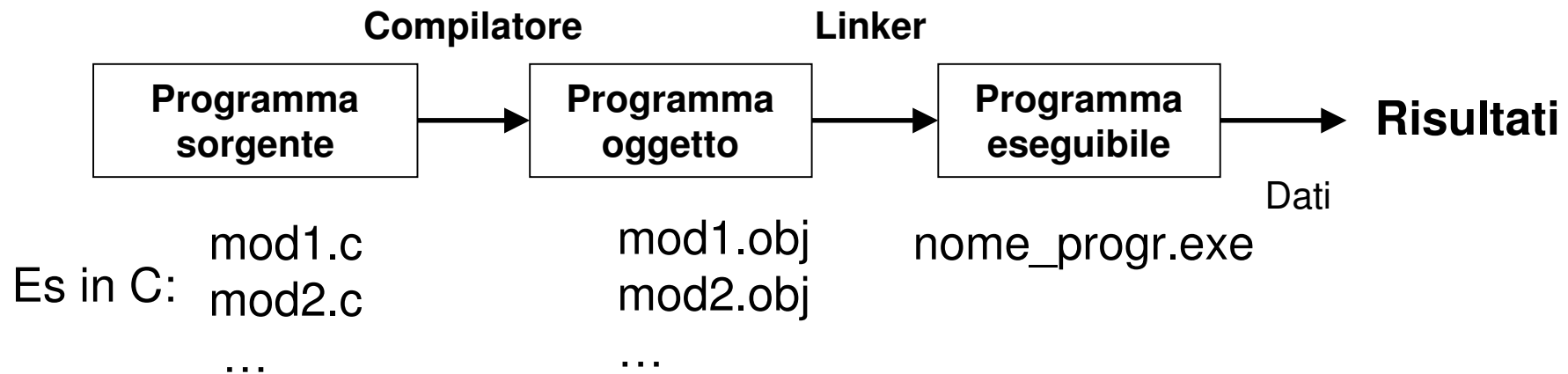
Linguaggio macchina

```
graph TD; A[Linguaggi ad alto livello (orientati all'uomo)] --> B[Compilatori o Interpreti]; B --> C[Linguaggio macchina]; D[Linguaggi a basso livello (linguaggi assemblativi, orientati alla macchina)] --> E[Assemblatori]; E --> C;
```

Schema di Compilazione

- Compilatore:

- riceve un intero programma sorgente e produce in un file l'intero programma oggetto.

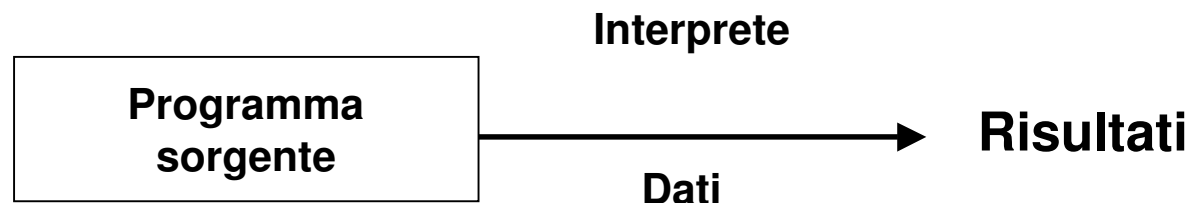


- Linker:

- Collega più moduli oggetto prodotti dal compilatore in un unico programma eseguibile;
- I riferimenti esterni ad ogni modulo non sono risolti dal compilatore
- Il linker risolve i riferimenti esterni.

Schema di Interpretazione

- Interprete:
 - Legge una singola frase in linguaggio sorgente, la trasforma in una sequenza di istruzioni macchina e le manda in esecuzione;
 - Traduzione e esecuzione sono contestuali.

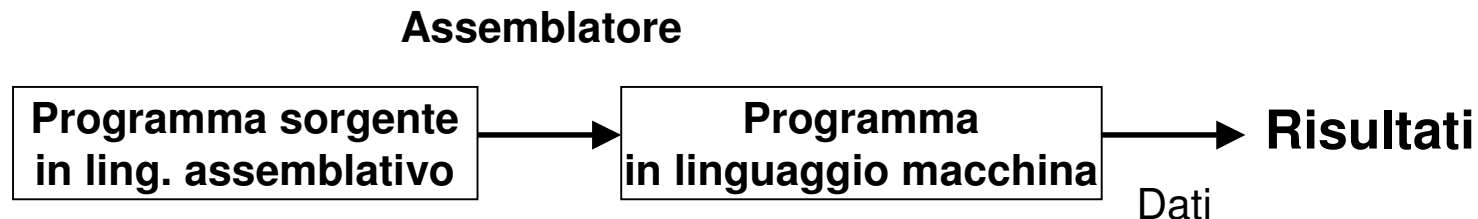


Compilatori e Interpreti: caratteristiche

- **Compilatori:**
 - Codice ottimizzato;
 - Programmi più veloci da eseguire;
 - I compilatori sono legati all'architettura dell'elaboratore tramite il sistema operativo.
- **Interpreti:**
 - Programmi più lenti (traduzione ed esecuzione sono contestuali);
 - Messa a punto del programma (*debugging*) migliore (si conosce la riga di programma dove si è verificato un errore);
 - Gli interpreti sono legati all'architettura dell'elaboratore tramite il sistema operativo.

Linguaggi Assemblativi e Assemblatori

- Un programma scritto in linguaggio macchina è una sequenza di istruzioni elementari codificate in binario;
- Scrittura dei programmi difficile (variabili legate ad indirizzi di macchina, ogni nuova istruzione inserita può determinare lo spostamento delle locazioni di memoria per ogni variabile);
- I linguaggi assemblativi sono linguaggi simbolici che usano simboli (nomi) per indicare il codice operativo e le variabili;
- Assemblatori: sono i traduttori dei linguaggi assemblativi;



Linguaggi e Metalinguaggi

- Un traduttore deve essere in grado di riconoscere le frasi come appartenenti al linguaggio che deve tradurre;
- Occorre definire un linguaggio mediante una sintassi (regole grammaticali) che consentono al traduttore di stabilire se una frase appartiene o meno al linguaggio;
- Sintassi: insieme delle regole che servono per determinare se un testo è strutturato correttamente;
- Semantica: insieme delle regole che consentono di dare un significato alle frasi del linguaggio;
- Metalinguaggio:
 - linguaggio che consente la definizione di linguaggi;
 - Insieme delle categorie sintattiche (es: soggetto verbo) e delle regole sintattiche (es: una frase nominale è composta da un soggetto seguito da un verbo).

Definizione di Grammatica

- $G = (N, T, P, S)$
 - $N = \underline{\text{Simboli Non Terminali}}$ (categorie sintattiche (articolo, nome, verbo));
 - $T = \underline{\text{Simboli Terminali}}$ (elementi di un vocabolario);
 - $P = \underline{\text{Produzioni}}$ (regole sintattiche, ogni produzione descrive le regole attraverso cui un simbolo non terminale è definito per mezzo di altri simboli non terminali e/o terminali);
 - $S = \underline{\text{Scopo}}$ (particolare simbolo non terminale).

Definizione di Linguaggio

$$L(G) = \{w \in T^* : S \Rightarrow w^*\}$$

- E' una definizione generativa di linguaggio;
- Un linguaggio, funzione di una grammatica, è l'insieme di tutte le frasi ottenute come combinazioni di simboli terminali, ammessi dalle regole di produzione che dallo scopo S generano le possibili frasi (w^*) del linguaggio.

Esempi di descrizione di “Produzioni”

- Le più usate modalità di descrizione di regole sintattiche (*Produzioni*) sono:
 - Notazione EBNF (Extended Backus-Naur Form);
 - Diagrammi sintattici.

Esempio di Grammatica

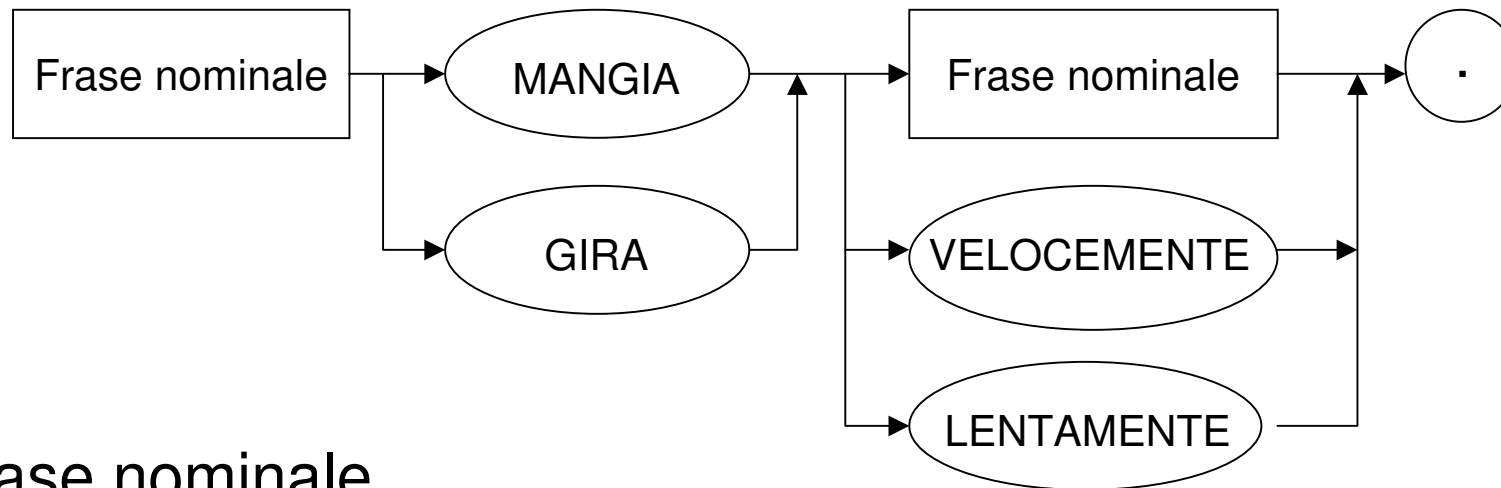
- S = frase;
- $T = \{\text{UN, GATTO, MANGIA, TOPO, VELOCEMENTE, GIRA, LENTAMENTE, BIANCO, IL, GRASSO, MAGRO}\};$
- $N = \{\text{frase, soggetto, predicato, frase-nominale, lista-di-aggettivi, articolo, nome, aggettivo, verbo, avverbio}\};$
- $P = [\text{regole sintattiche descritte secondo “EBNF” o “Diagrammi Sintattici”}].$

Produzioni descritte mediante EBNF

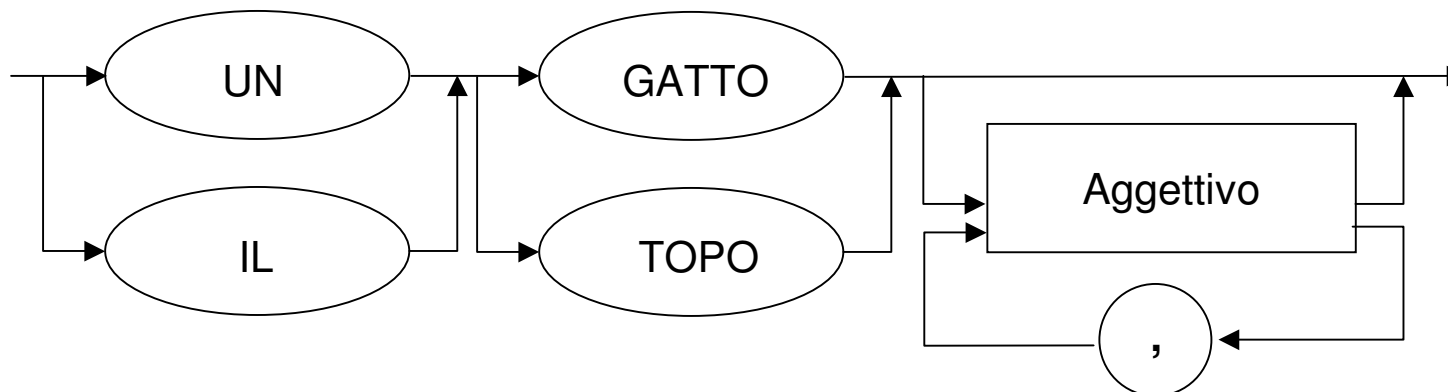
1. Frase = soggetto predicato “.”
 2. Soggetto = frase-nominale
 3. Predicato = verbo (frase-nominale | avverbio)
 4. Frase-nominale = articolo nome [lista-di-aggettivi]
 5. Lista-di-aggettivi = aggettivo {“,” aggettivo}
 6. Articolo = “UN” | “IL”
 7. Nome = “GATTO” | “TOPO”
 8. Aggettivo = “NERO” | “BIANCO” | “MAGRO” | “GRASSO”
 9. Verbo = “GIRA” | “MANGIA”
 10. Avverbio = “VELOCEMENTE” | “LENTAMENTE”
- (| = oppure; { } = ripetizione; () = raggruppano alternative; [] = parte opzionale)

Produzioni descritte mediante Diagrammi Sintattici

frase



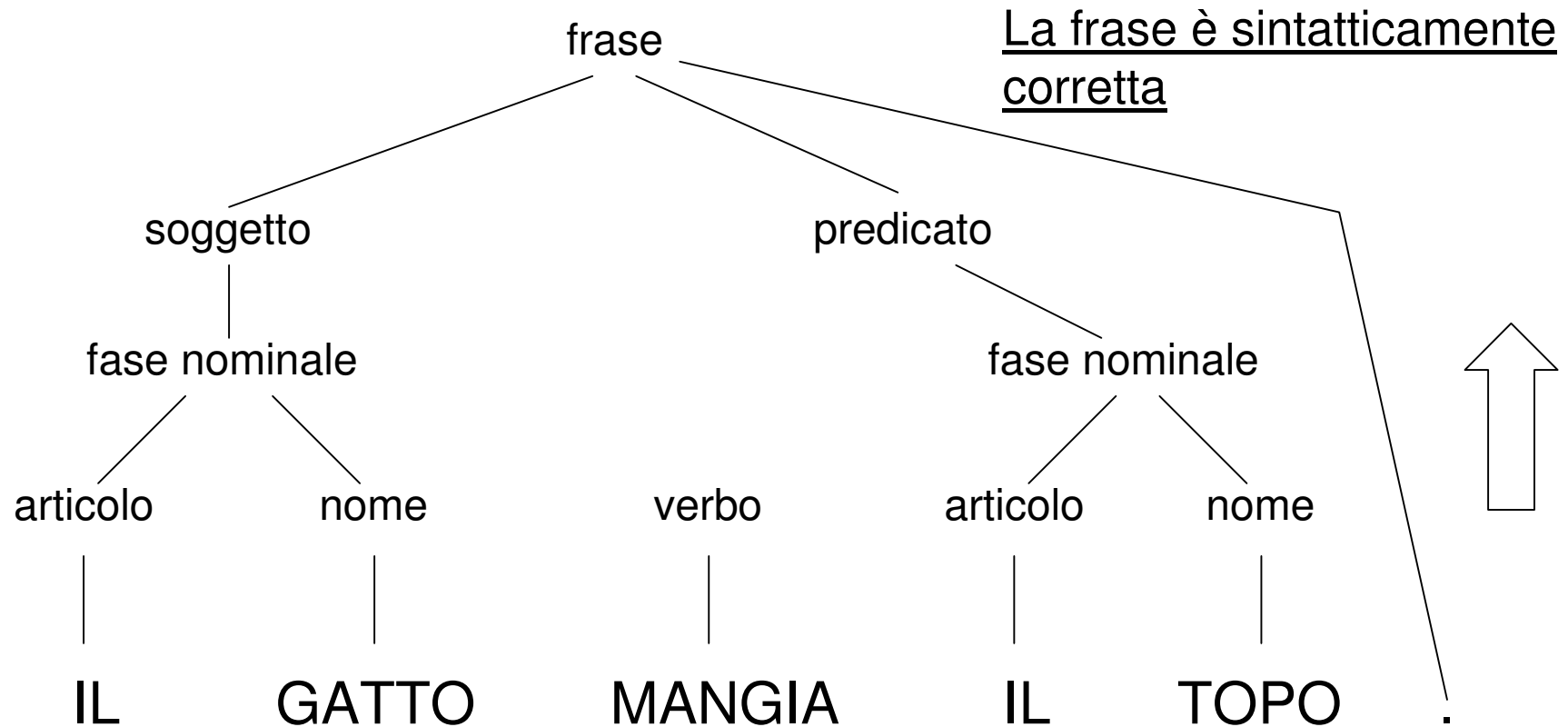
frase nominale



Analisi Sintattica

- Le regole sintattiche possono essere usate:
 1. Per costruire frasi sintatticamente corrette;
 2. Per riconoscere se una frase è sintatticamente corretta.

Esempio di Analisi Sintattica Bottom Up



Struttura dei Linguaggi di Programmazione

- Rappresenta un algoritmo e specifica l'insieme di operazioni da applicarsi in una certa sequenza su un certo insieme di dati;
- Occorre definire:
 - Caratteristiche dei dati;
 - Modalità di accesso ai dati;
 - Operazioni sui dati;
 - Ordine di esecuzione delle operazioni.
- Ogni dato deve essere descritto rispetto al tipo e alla struttura:
 - Tipo: definisce l'insieme delle operazioni ammissibili;
 - Struttura: relazioni esistenti tra i valori che rappresentano il dato (matrici, vettori, strutture (record));
- Un programma è composto di:
 - Commenti;
 - Frasi dichiarative (direttive al traduttore, ad esse non corrispondono istruzioni di macchina (Dichiarazione di variabili e funzioni));
 - Frasi esecutive (azioni da applicare ai dati, corrispondono ad istruzioni di macchina).