Appunti di
Linguaggi e Traduttori

A cura di:
Francesco Refolli
Matricola 865955

Anno Accademico 2024-2025

Chapter 1

28/02/25

1.1 Stutture Dati

Le strutture dati utilizzate sono gli alberi, i grafi, le pile, le code.

1.2 Algoritmi

1.2.1 Visita Pre

E' un processo di ereditarieta' => $attr(child) = compute(attr(root))$.

```
def preorder(tree, visit):
  root, children = tree
  visit(root)
  for child in children:
    preorder(child, visit)
```

1.2.2 Visita Post

E' un processo di sintesi => $attr(root) = sum([attr(child) for child in children])$.

```
def postorder(tree, visit):
  root, children = tree
  for child in children:
    preorder(child, visit)
  visit(root)
```

1.2.3 Visita Level

```
def levelorder(tree, visit):
  Q = Queue([])

  Q.enqueue(tree)
  while len(Q) > 0:
    tree = Q.dequeue()
    root, children = tree
    visit(root)
    for child in children:
      Q.enqueue(child)
```

1.3 Grafi

Diagramma di ASSE

1.3.1 Depth First

```
def depthfirst(adiacency, start, visit):
  S = Set({})
  def recursive(src):
    S.add(src)
    visit(src)
    for dst in adiacency[src]:
      if dst not S:
        recursive(dst)
```

1.3.2 Breadth First

```
def breadthfirst(adiacency, start, visit):
  S = Set({})
  Q = Queue({start})
  while len(Q) > 0:
    src = Q.dequeue()
    S.add(src)
    visit(src)
    for dst in adiacency[src]:
      if dst not S:
        Q.enqueue(dst)
```

1.3.3 BackTracking

Abilita' di rendersi conto che il percorso in profondita' non e' ottimale e quindi tornare sui miei passi per processare altri percorsi.

```
def backtrack(candidate):
  if reject(candidate):
    return
  if accept(candidate):
    output(candidate)
  s = first(candidate)
  while s:
    backtrack(s)
    s = next(candidate)
```

Chapter 2

04/03/2025

2.1 Linguaggi e grammatiche

Sia T l'insieme dei simboli (*alfabeto*), il linguaggio $L \subset T^*$. Ci interessano i linguaggi infiniti.

Le descrizioni finite (*grammatica*) non possono descrivere tutti i linguaggi.

Mi immagino il linguaggio delle parole $L = w_i \mid w_i \notin L_i$. Non esiste una descrizione in grado di catturare tutto (ed esattamente) il linguaggio.

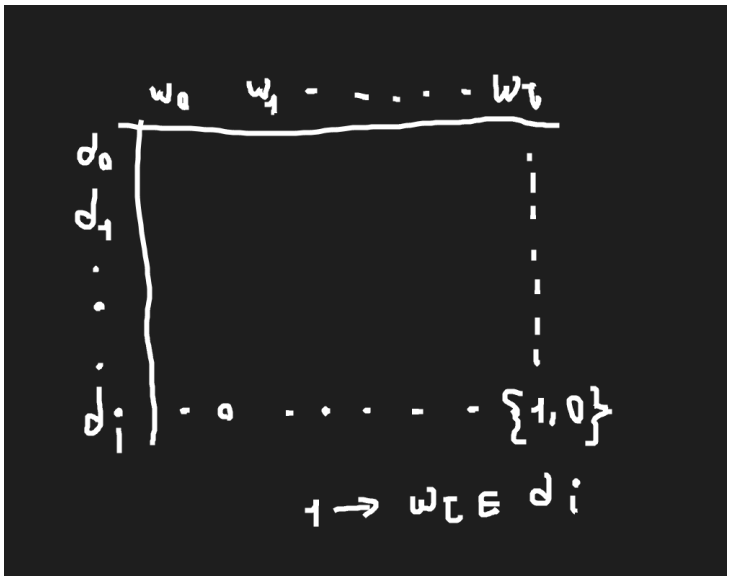

Figure 2.1: Si immagina la matrice delle grammatiche e delle parole, dove la cella (ij) rappresenta se la parola j-esima appartiene al linguaggio della descrizione i-esima

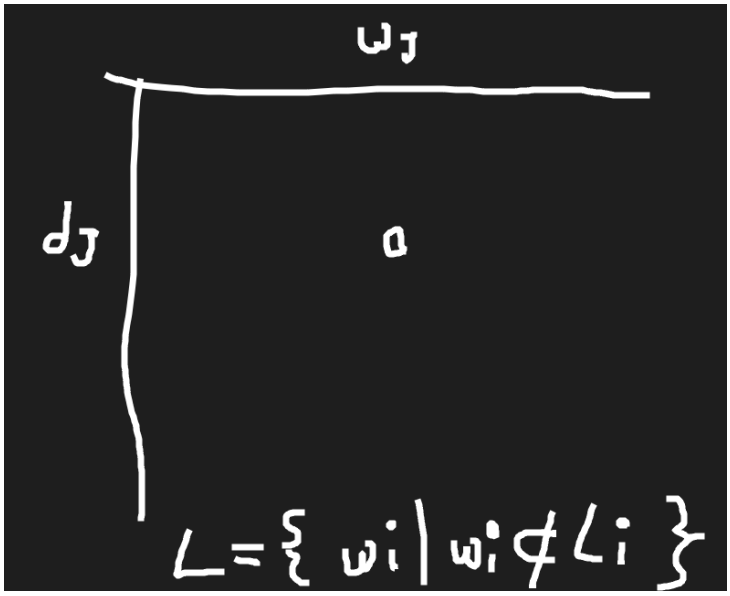

Figure 2.2:

2.2 Grammatiche

Le grammatiche sono pensate come generatore del linguaggio. Si puo' invertire la relazione e utilizzare le grammatiche per riconoscere le parole del linguaggio. Sia T l'insieme dei simboli terminali. Sia N l'insieme dei simboli non terminali (la generazione non e' terminata quando si ottengono, possono essere soggetti ad ulteriori trasformazioni). Ex: $T = a, b,\ N = S, U, C, A$. Sia $P = (T \cup N)^+ \to (T \cup N)^*$ l'insieme delle regole di produzione. Se $w_0 \to w_1 \to w_2 ... \to w_n$ allora si dice che $w_0 \Rightarrow w_n$. Uno dei simboli non terminali, S, viene identificato come il simbolo iniziale. Una grammatica $G = (T, N, P, S)$. Il linguaggio descritto dalla grammatica G e' $L = w \in T^* \mid S \Rightarrow w$.

2.3 Gerarchia di Chomsky

In base alla forma delle regole di P, si determinano dei sottoinsiemi di linguaggi catturabili. Se un linguaggio e' riconoscibile con una grammatica di tipo K non significa che non possa essere necessariamente riconosciuta con una grammatica di tipo $K + 1$. Vice versa se non e' riconoscibile con una grammatica di tipo K allora non e' riconoscibile con una grammatica di tipo $K + 1$.

- Tipo 0: va bene tutto, non ci sono restrizioni.
- Tipo 1:
  - (context sensitive), produzioni del tipo $\alpha\sigma_0\beta \to \alpha\sigma_1\beta$. Le parole del "contesto" rimangono uguali.
  - (monotone), produzioni dove $|sigma_0| \leq |sigma_1|$. Succede che per produrre parole di lunghezza N uso necessariamente più regole che per produrre parole di lunghezza $K < N$.

- Tipo 2: (context free), produzioni del tipo $N \to (N \cup T)^*$. (Formalmente c'e' un problema con le derivazioni nulle, con ϵ, perche' cosi' non sono monotone, ma la linea ufficiale è che ce ne sbattiamo).
- Tipo 3: (espressioni regolari), produzioni del tipo $N \to T\ N \mid N\ T\ T\ T$.