linguaggi di programmazione

Matteo

December 19, 2025

ricorda sul libro per modulo 1 capitoli da 1 a 5

1 intro

linguaggi imperativi , linguaggi dichiarativi, funzionali e logici linguaggi imperativi

- basato sulla nozione di stato, le istruzioni sono comandi che cambiano lo stato
- stato= insieme di locazioni di memoria contenente valori
- generalmente di basso livello

linguaggi dichiarativi

- basati sulla nozionne di funzioni o relazione
- le istruzioni sono dichiarazioni di nuovi valori

linguaggi funzionali

- basati sullla nozione di funzione: risultato di un programma = valore esplicito di una espressione
- ricorsione
- programmare= costruire una funzione

linguaggi logici

- basati sulla nozione di relazione: risultato di un programma = insieme di valori di variabili determinato da relazioni
- istruzioni = implicazioni logiche fra opportune formule , che possono essere viste come regole di riscrittura
- programmare= definire la relazione che definisce il valore delle variabili di interesse

linguaggi orienatati agli oggetti: ora molto usati, sono linguaggi imperativi con alcune ,etodologie dichiarative. Oggetti (istanze di opportune classi) che contengono i dati (concetti imperativi) e metodi come funzioni per operare su tali oggetti (concetti dichiarativi).

2 Macchine astratte

pag 28 :Ml molto complesso da tradurre in una macchina fisica ad alto livello, quindi vengono costruite macchine fisiche solo per linguaggi di basso livello

altimenti crei u programma che trduce L e i suoi costrutti in L', un linguaggio gia esistente implementabile su una macchina gia esistente; minore velocita rispetto al casoo precedente

altrimenti anziche programmi usi microprogrammi o firmware che traducono L in un liguaggio di basso livello e usano registri di sola lettura, garantendo buone prestazioni

pag 31 impl. inter. pura : non e vvera traduzione, I fa corrispondere a una certa parte di L una certa parte di Lo, poco efficente, I deve decodificare al momento. piu flessibile perche permette di interagire direttamnete con l'esecuzione del programma e gli interpreti sono piu veloci da creare

impl. comp. pura: vera traduzione dal compilatore che avviene prima dell'esecuzione del programma. alcune informazioni del programma sorgente vanno perse che rende piu difficile interagire con il programma in tempo reale

vantaggi e svantaggi interpretazione:

- vantaggio: maggiore flessibilita, permette di interagire con l'interazione del programma.
- vantaggio : piu veloce da realizzare
- vantaggio : occupa meno memorio, non generando nuovo codice, problema meno sentito oggi.
- svantaggio : la compilazione intterpretativa e meno efficente perche deve effettuare al momento dell'esecuzione un'interpretazione dei costrutti di L. con diverse occorrenze dello stesso costrutto si richiedono ulteriori decodifiche ogni volta

vantaggi e svantaggi compilazione:

- vantaggio: il compilatore deve compilare una sola volta al"inizio poi i costrutti non devono essere decodificati ogni volta
- vantaggio: maggiore efficenza
- savntaggio: perdita del codice sorgente. se ci fosse un problema sarebbe difficile capire da dove si origiina l'errore

siamo a pag 35 di fatto nei linguaggi reali sono presenti entrambe le tecniche. ad esempio abbiamo linguaggio L, compilato in linguaggio intermedio Li, interpretato in lo se l'interprete della macchina intermmedia e molto diverso dal linguaggio finale Lo diremo che e un implementazione interpretativa, se e quasi uguale e un implementazione compilatiiva.

la differnza tr acompletamente interpretattiva e mostly interpretattiva e che non tutti i costrutti devono essere simulati

per impl. mostly compilativa alcune funzionalita devono essere simulate perche Li non trova un corrispondente immediato in molto

ci sono tanti spettri intermedi, di solito si predilige flessibilita

di solito ci sono piu livelli di macchine astratte con i loro linguaggi e funzionalita. praticamente tutto nell'informatica e creato attravero una gerarchi adi macchine astratte(programmi e linguaggi).

ogni livello accede a quello inferiore e aggiunge nuove funzionalita, in un certo senso creando un nuo liguaggioMnLn

cio peremtte un certo dominio sulla complessita del sistemaa e independenza tra i livelli

macchina astratta: una formalizzazione astratta di un generico esecutore di algoritmi formalizzato attraverso un linguaggio di programmazione

interprete: un componente essenziale della macchina astratta che ne caratterizza il comportamento, mettendo in relazione ""operazionale" il linguaggio della macchina astratta col mondo fisico circostante

3 descrivere i linguaggi di programmazione

secondo morris i linguaggi hanno tre campi che li descrivono:

- **grammatica**: indica quali frasi sono corrette, i lessico si occupa delle singole parole come sequenze di segni, la sintassi delle frasi come sequenze di parole
- **semantica**:risponde alla domanda "cosa significa una frase corretta", per quanto riguarda i linguaggi artificiali, si tratta del loro scopo
- **pragmatica**:risponde alla domanda " come usare una frase corretta e sensata", si occupa dell'uso della frase rispetto al contesto

per quanto riguarda i linguaggi di programmazione si puo parlare anche di implementazione cio mediante quale processo le frasi "operative" del linguaggio realizzano il loro fine.

3.1 grammatica e sintassi

tecniche generative: espedienti trovati nell'ambito dei linguaggi naturali per limitare nelle grammatiche le ambiguita dei linguaggi naturali non troppo utili per i ling. naturali, ma utilissimi nei ling. di programmazione

3.1.1 grammatiche libere

definito un insieme finito di elementi come **alfabeto**, dato un alfabetoA, e dato A*= insieme di stringhe finite di A. un linguagggio e un sottoinsieme di A*, una grammatica formaale serve a definire un di stringhe tra quelle possibili su un dato alfabeto

grammatica libera da conteso composta sa 4 parti:

- simbolii non terminali: insieme finito di simboli astratti che non compaionon nel concreto
- simboli terminali: insieme finito di simboli concreti che compaiono nelle stringhe del linguaggio
- produzioni: le regole di riscrittura che permettono di sostituire un simbolo non terminale con una sequenza di simboli terminali e non terminali

- simbolo iniziale da cui parte tuttoù