A lexikális elemzés

Fordítóprogramok előadás (A,C,T szakirány)

A lexikális elemzés helye

Elemzési lépések szétválasztása

- lexikális elemzés: megadható reguláris nyelvtannal (3-as)
- szintaktikus elemzés: megadható környezetfüggetlen nyelvtannal (2-es)
- szemantikus elemzés: környezetfüggő (1-es)
  - pl. egy változó használatának helyessége függhet a deklarációjától, azaz a környezettől

A három lépés szétválasztásának oka, hogy egyszerű feladathoz ne használjunk bonyolult eszközöket.

Reguláris nyelvtan

Reguláris nyelvtanokban (Chomsky 3) a szabályok a következő alakúak lehetnek:

Jobbrekurzív eset

$A \to a$
$A \to aB$
$A \to \epsilon$

Balrekurzív eset

$A \to a$
$A \to Ba$
$A \to \epsilon$

Reguláris nyelvtan

Reguláris nyelvtanokban (Chomsky 3) a szabályok a következő alakúak lehetnek:

Jobbrekurzív eset

$A \to a$
$A \to aB$
$A \to \epsilon$

Balrekurzív eset

$A \to a$
$A \to Ba$
$A \to \epsilon$

Példa: változónevek leírása

$V \to \underline{a}\ F \mid \underline{b}\ F \mid \ldots$
$F \to \epsilon \mid \underline{a}\ F \mid \underline{b}\ F \mid \ldots \mid \underline{1}\ F \mid \underline{2}\ F \mid \ldots$

Reguláris kifejezés

- kifejezőereje azonos a reguláris nyelvtanokéval
- alapelemek:
  - üres halmaz
  - üres szöveget tartalmazó halmaz
  - egy karaktert tartalmazó halmaz
- konstrukciós műveletek:
  - konkatenáció
  - unió: |
  - lezárás: *
- további „kényelmi" műveletek: +, ?

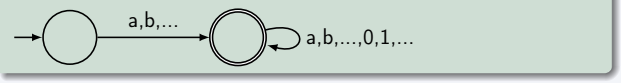
Példák (flex szintaxissal)

változónév: `[a-zA-Z][a-zA-Z0-9]*`
egész szám: `(\+|\-)?[0-9]+`
törtszám: `[0-9]+''.''[0-9]*`

Véges determinisztikus automaták

- kifejezőereje azonos a reguláris nyelvtanokéval és a reguláris kifejezésekével
- elemei:
  - ábécé
  - állapotok halmaza
  - átmenetfüggvény
  - kezdőállapot
  - végállapotok halmaza

Változónevek elfogadása automatával



Véges determinisztikus automata implementációja

- egymásba ágyazott `if` vagy `case` utasításokkal egy cikluson belül
  - elágazunk a pillanatnyi állapot szerint
  - ezen belül elágazunk a következő karakter szerint
  - az egyes ágakban beállítjuk a következő állapotot és kezdjük előről
- táblázattal
  - a táblázat soraihoz az állapotok, oszlopaihoz a karakterek vannak rendelve
  - celláiban a következő állapot sorszáma van
  - a következő állapot a pillanatnyi állapot sorában és az olvasott karakter oszlopában található

A lexikális elemző működése

- Az `abc` bemenet esetén `a`, `ab` és `abc` is legális változónév. Melyiket kellene felismerni?
  - A lexikális elemző mindig a lehető leghosszabb karaktersorozatot ismeri fel.

A lexikális elemző működése

- Az `abc` bemenet esetén `a`, `ab` és `abc` is legális változónév. Melyiket kellene felismerni?
  - A lexikális elemző mindig a lehető leghosszabb karaktersorozatot ismeri fel.
- A `while` input megfelel a változónév definíciójának és egy kulcsszó is egyben. Melyiket kellene felismerni?
  - A lexikális elemző megadásakor sorbarendezhetjük a szimbólumok definícióit. Ha egyszerre több szimbólum is felismerhető, a sorrendben korábbi lesz az eredmény. (Tehát a kulcsszavak definícióját kell előre venni.)

Kulcsszavak és standard szavak

Kulcsszó

A kulcsszavaknak előre adott jelentésük van, és ez nem definiálható felül.

Kulcsszavak és standard szavak

Kulcsszó

A kulcsszavaknak előre adott jelentésük van, és ez nem definiálható felül.

Standard szó

A standard szavaknak előre adott jelentésük van, de ez felüldefiniálható.

Kulcsszavak és standard szavak

Kulcsszó

A kulcsszavaknak előre adott jelentésük van, és ez nem definiálható felül.

Standard szó

A standard szavaknak előre adott jelentésük van, de ez felüldefiniálható.

- Ha a kulcsszavakat is véges determinisztikus automatával akarjuk felismerni, nagyon nagy méretű automatát kaphatunk.
- Jobb módszer egy táblázatban tárolni őket: akárhányszor a lexikális elemző egy azonosítót ismer fel, meg kell nézni, hogy benne van-e ebben a táblázatban. (Ha igen, akkor kulcsszó, különben azonosító.)

Előreolvasás

A lexikális elemző időnként több karaktert is előreolvas a szimbólum felismeréséhez.

Előreolvasás

A lexikális elemző időnként több karaktert is előreolvas a szimbólum felismeréséhez.

1. példa: az egyes szimbólmok egymás prefixei
   - egész szám: [0-9]+
   - valós szám: [0-9]+"."[0-9]+
   - 3.14 ⇒valós szám
   - 3.x ⇒egész szám, majd lexikális hiba
   - az elemző megjegyzi a legutóbbi érvényes állapotot

Előreolvasás

A lexikális elemző időnként több karaktert is előreolvas a szimbólum felismeréséhez.

1. példa: az egyes szimbólmok egymás prefixei
   - egész szám: [0-9]+
   - valós szám: [0-9]+"."[0-9]+
   - 3.14 ⇒valós szám
   - 3.x ⇒egész szám, majd lexikális hiba
   - az elemző megjegyzi a legutóbbi érvényes állapotot
2. példa: Fortran (a szóközöknek semmi szerepe nem volt!)
   - `DO 10 I = 1.1000` (ez egy értékadás a `DO10I` változónak)
   - `DO 10 I = 1,1000` (ez egy ciklus)

Előreolvasás

A lexikális elemző időnként több karaktert is előreolvas a szimbólum felismeréséhez.

1. példa: az egyes szimbólmok egymás prefixei
   - egész szám: [0-9]+
   - valós szám: [0-9]+"."[0-9]+
   - 3.14 ⇒valós szám
   - 3.x ⇒egész szám, majd lexikális hiba
   - az elemző megjegyzi a legutóbbi érvényes állapotot
2. példa: Fortran (a szóközöknek semmi szerepe nem volt!)
   - `DO 10 I = 1.1000` (ez egy értékadás a `DO10I` változónak)
   - `DO 10 I = 1,1000` (ez egy ciklus)
   - megoldás: `DO / [0-9]+ [a-zA-Z0-9]* = [a-zA-Z0-9]*` ,
     - a '/' az előreolvasási operátor
     - *r*/*s* jelentése: ismerd fel *r*-t, de csak ha *s* követi
     - *r* felismerése után *s* visszakerül a bemenetbe
     - a leghosszabb karaktersorozatról való döntéskor | *r* | + | *s* | számít

Szemantikus értékek és szimbólumtábla

- A lexikális elemzőnek a felismert szimbólum fajtáján kívül egyéb információkat is továbbítania kell.
  - változó: a változó neve
  - konstans: a konstans értéke
- Ezekre a szemantikus értékekre szemantikus elemzéshez és a kódgeneráláshoz van szükség.
- A változókat és azok adatait a szimbólumtáblába kell felírni.
  - Ezt általában a szintaktikus elemző teszi meg a változó deklarációjának felismerésekor.

Direktívák

Példa direktívákra

```
#include ''my.h''
#define valami 42
#ifndef FELTETEL
int akarmi() { return valami; }
#endif
```

Célszerű egy előfeldolgozó fázis beiktatása a lexikális és a szintaktikus elemzés közé, ami

- feljegyzi a makródefiníciókat,
- elvégzi a makróhelyettesítéseket,
- meghívja a lexikális elemzőt a beillesztett fájlokra,
- kiértékeli a feltételeket és dönt a kódrészletek beillesztéséről vagy törléséről.

Hibatípusok és javítási lehetőségek

- illegális karakter (pl. `add?ress`, `?` legyen az illegális karakter)
  - az éppen épített szimbólum eldobása és folytatás a következő karaktertől (eredmény: `ress` azonosító)
  - a karakter kihagyása (erdemény: `address` azonosító)
  - a karakter helyettesítése szóközzel (eredmény: `add` és `ress` azonosítók)

Hibatípusok és javítási lehetőségek

- illegális karakter (pl. `add?ress`, `?` legyen az illegális karakter)
  - az éppen épített szimbólum eldobása és folytatás a következő karaktertől (eredmény: `ress` azonosító)
  - a karakter kihagyása (erdemény: `address` azonosító)
  - a karakter helyettesítése szóközzel (eredmény: `add` és `ress` azonosítók)
- elgépelt kulcsszó (pl. `while` helyett `wile` `whille` `wjile`)
  - a lexikális elemző azonosítónak fogja felismerni, de egy ügyes szintaktikus elemző kijavíthatja a hibát
  - azokban a nyelvekben, ahol a kulcsszavakat speciális módon jelölik, a lexikális elemző is felismerheti és javíthatja a hibát
    - speciális jelölés lehet pl. adott karakterrel való kezdés, zárójelezés, nagybetűk használata
    - a szintaxiskiemelés (pl. `vastagítás`, `színezés`) nem használható erre a célra, mert az láthatatlan a lexikális elemző számára!

Hibatípusok és javítási lehetőségek

- kihagyott szimbólum (pl. `1+a` helyett `1a`, `a+1` helyett `a1`)
  - ezeket csak a szintaktikus elemző tudja észrevenni

Hibatípusok és javítási lehetőségek

- kihagyott szimbólum (pl. `1+a` helyett `1a`, `a+1` helyett `a1`)
  - ezeket csak a szintaktikus elemző tudja észrevenni
- hibás számformátum (pl. `1.23.45`)
  - valamelyiket illegális karakternek lehet tekinteni

Hibatípusok és javítási lehetőségek

- kihagyott szimbólum (pl. `1+a` helyett `1a`, `a+1` helyett `a1`)
  - ezeket csak a szintaktikus elemző tudja észrevenni
- hibás számformátum (pl. `1.23.45`)
  - valamelyiket illegális karakternek lehet tekinteni
- befejezetlen megjegyzések és sztringek (pl. `''alma ...`, `/* megjegyzés ...`)
  - könnyen az egész további program megjegyzésbe kerülhet
  - sor végén, illetve fájl végén lehet jelezni a hibát