Processadors de Llenguatges

Pràctica II – Eines de suport a la generació d'analitzadors sintàctics

Curs 22/23

Objectius

L'interès de la pràctica és que l'alumne formalitzi les regles sintàctiques per a llenguatges formals i utilitzi una eina de generació automàtica d'analitzadors sintàctics basats en la tècnica LR per a la seva implementació.

Podeu utilitzar l'eina de generació automàtica d'analitzadors sintàctics que més us agradi: *lex i yacc* o *flex i bison* per generar codi C; *JFlex i CUP* o *JFlex i BYacc/J* per generar codi Java; *PLY* per generar codi Python.

Exercici 1. Traducció de notació CNF DIMACS a notació clausal

En aquest exercici heu d'implementar un traductor de notació CNF DIMACS a notació clausal.

A mode de recordatori un fitxer en format CNF DIMACS té la sintaxi següent:

```
c Exemple CNF DIMACS: mini-sudoku de 2 x 2
c CNF amb 4 variables i 6 clàusules
c  Final de clàusula: 0
p cnf 4 6
1 2 0
-1 -2 0
3 4 0
-3 -4 0
-1 -3 0
-2 -4 0
```

Que codifica les restriccions en notació clausal d'un mini-sudoku de 2 x 2:

$p1 \vee p2$
$\neg p1 \vee \neg p2$
$p3 \vee p4$
$\neg p3 \vee \neg p4$
$\neg p1 \vee \neg p3$
$\neg p2 \vee \neg p4$

Es demana:

- Implementeu una especificació lèxica i sintàctica que donat un fitxer en format CNF DIMACS generi el fitxer equivalent en notació clausal.
- El final de cada línia del fitxer vindrà indicat pel caràcter new line i el final de tractament per la marca de final d'arxiu EOF.
- Desprecieu els caràcters blancs i tabuladors existents a l'entrada.
- L'entrada podrà contenir comentaris indicats amb el caràcter `c` i acabats en new line. Desprecieu els comentaris a nivell lèxic.
- Incorporeu el tractament d'errors, seguint l'estratègia del mode pànic, a nivell sintàctic. Els possibles errors lèxics seran tractats a nivell sintàctic. Indiqueu el número de línia on es detecta l'error.
- Verifiqueu l'error semàntic: el nombre de clàusules i variables especificat és correcte.

Exercici 2. Mini calculadora amb expressions condicionals
Implementeu una especificació lèxica i sintàctica que simuli una petita calculadora. La calculadora tindrà 26 registres de tipus enter etiquetats en el rang `[a-z]` i acceptarà expressions condicionals.

Característiques de la calculadora:

- Les instruccions que ha de suportar la mini calculadora són 3:
  1. Avaluació d'expressions enters.
  2. Assignació de valors als registres. Utilitzarem l'assignació, operador =, per assignar valors als registres, els quals podran ser utilitzats en posteriors expressions.
  3. Ús d'expressions condicionals. Incorporar les expressions condicionals amb una sintaxi de la forma:

     ```
     expressió_lògica ? expressió1 : expressio2;
     ```

     on `expressió1` i `expressio2` denoten l'avaluació d'una expressió o l'assignació d'un valor a un registre i `expressio_lògica` és una expressió lògica (Boolean).
- Els operadors aritmètics, lògics i de relació mínims que heu de considerar són els següents (considereu els formats lèxics que vulgueu):
  - Operadors aritmètics binaris d'enters: $+$, $-$, $*$, $/$, % (residu).
  - Operadors aritmètics unaris d'enters: $+$, $-$ (canvi de signe unari).
  - Operadors de relació: $<$, $>$, $<=$, $>=$, $==$, $!=$.
  - Operadors lògics: !, &&, ||.
  - Considereu la possibilitat de tractar altres operadors. Per exemple els operadors de desplaçament de bits (únicament definits per als enters): $<<$, $>>$.
    També podeu incorporar altres operadors com els de manipulació de bit (també únicament definits per als enters): ~ (complement a u), & (AND entre bits), | (OR inclusiu entre bits), ^ (OR exclusiu entre bits).

- Els operands poden ser registres i constants enteres amb els formats lèxics que trieu.
- El final d'una expressió vindrà indicat pel caràcter ';' i el final de tractament per EOF.
- Un cop calculat el resultat d'una expressió l'escriurem a la sortida estàndard, excepte si es tracta d'una expressió d'assignació. En aquest cas, actualitzarem el valor del registre referenciat.
- En el cas de les expressions condicionals, es mostrarà el resultat de l'`expressió1` o `expressio2`, depenent del resultat d'avaluar l'`expressio_lògica`.
- Desprecieu els caràcters blancs, tabuladors i nova línia existents a l'entrada.
- Definir un format per a incloure línies de comentaris. Ignorar els comentaris existents a l'entrada.
- Verifiqueu que el llenguatge acceptat per l'analitzador sintàctic és el desitjat. És a dir, verifiqueu que elimineu tots els conflictes mitjançant la definició, per a cada operador, del corresponent nivell de precedència i associativitat. Normalment, de major a menor precedència tenim la relació següent:

Canvi de signe (unari)	+ -
Producte	*, /, %
Suma i diferència (binari)	+ -
Desplaçament de bit	<< >>
Operadors relacionals	< > <= >=
Equivalència	== !=
Complement bit a bit	~
AND entre bits	&
OR exclusiu entre bits (XOR)	^
OR inclusiu entre bits	|
NOT lògic	!
AND lògic	&&
OR lògic	||

- Incorporeu el tractament d'errors, seguint l'estratègia del mode pànic, a nivell sintàctic. Els possibles errors lèxics seran tractats a nivell sintàctic. Indiqueu el número de línia on es detecta l'error.
- Al concluir el processat es mostrarà el valor final dels registres definits.

Per exemple, amb l'entrada següent:

```
// Assignació de valors
a = 2;
b = -3;
// Expressió condicional
(a * -b -2 >= 0 && b < 0) ?  c = a+b :     c = b;
 // Avaluació d'expressions aritmètiques
a-b;
c*2;
// Expressió condicional
(c %2 == 0 && a - b < 0) ?
     d = 1  :
     d = -1;
```

La sortida esperada és la següent:

```
5
-6
a = 2
b = -3
c = -3
d = -1
```

Exercici 3. Eliminar les implicacions i dobles implicacions al llenguatge CP0

Implementeu una especificació lèxica i sintàctica que accepti com a entrada una fòrmula proposicional definida sobre les lletres en majúscula (rang `[A-Z]`) i generi com a sortida la fòrmula proposicional equivalent sense les connectives d'implicació i doble implicació.

- A nivell lèxic considerarem l'especificació definida a l'Exercici 3 (Llenguatge de la lògica proposicional) de la 1a pràctica del curs.
- Les fòrmules proposicionals a analitzar estaran en un fitxer de test. Proporcioneu el nom del fitxer d'entrada com millor us vagi depenent del sistema escollit.
- El caràcter nova línia actuarà com a marca de final de fòrmula proposicional i la marca de final d'arxiu EOF com a marca de fi d'entrada.
- Desprecieu els caràcters blancs i tabuladors existents a l'entrada.
- La sortida la podeu escriure directament a la sortida estàndar o a un fitxer.
- Definir un format per a incloure línies de comentaris. L'entrada podrà contenir comentaris.
- Verifiqueu que el llenguatge acceptat per l'analitzador sintàctic és el desitjat. És a dir, verifiqueu que elimineu tots els conflictes mitjançant la definició, per a cada operador lògic, el corresponent nivell de precedència i associativitat. Recordeu que de major (1) a menor (4) prioritat tenim els operadors lògics següents:

  1. la negació !,
  2. la conjunció $\wedge$,
  3. la disjunció $\vee$ i
  4. la implicació representada amb el string "$->$", i la doble implicació representada amb el string "$<->$".

  A més, la negació ! és un operador unari associatiu per la dreta i, la resta són binaris i associatius per l'esquerra. D'altra banda, els parèntesis permeten modificar la prioritat associada amb els operadors.

- Incorporeu el tractament d'errors, seguint l'estratègia del mode pànic, a nivell sintàctic. Els possibles errors lèxics seran tractats a nivell sintàctic. Indiqueu el número de línia on es detecta l'error.

- Per transformar les fòrmules recordeu:

$$A \to B \equiv \neg(A) \vee (B)$$

$$A \leftrightarrow B \equiv (A \to B) \wedge (B \to A) \equiv (\neg(A) \vee (B)) \wedge (\neg(B) \vee (A))$$

  Per exemple, en el nostre llenguatge la fòrmula següent:

$$P \wedge Q <- > (!R \vee (Q- > T));$$

  pot ser transformada en la fòrmula equivalent següent:

$$(!(P \wedge Q) \vee ((!R \vee (!(Q) \vee (T))))) \wedge (!((!R \vee (!(Q) \vee (T)))) \vee (P \wedge Q))$$

- Per resoldre el problema, no cal construir l'arbre sintàctic a memòria i recorre'l avaluant les transformacions. Podeu anar construïnt la fòrmula equivalent com un string que es va completant amb el processament de l'entrada. Un cop acaba la fòrmula (fòrmules acabades en ;) es copia el string resultat a la sortida estàndard.

Exercici 4. Anàlisi de fòrmules de CP1

A l'exercici 4 de la primera pràctica vam implementar un analitzador lèxic per a fòrmules de CP1. Ara hem d'implentar l'analitzador sintàctic de manera que reconegui les fòrmules de CP1 ben formades.

Formalment, les fòrmules ben formades en CP1 es defineixen de la forma següent :

- **Termes:**
  1. Les variables i constants són termes.
  2. Si f és un símbol de funció i $t_1, \ldots, t_n$ són termes, aleshores $f(t_1, \ldots, t_n)$ és un terme.
- **Fòrmula atòmica:** Si P és un símbol de predicat i $t_1, \ldots, t_n$ són termes, aleshores $P(t_1, \ldots, t_n)$ és una fòrmula atòmica.
- **Les fòrmules ben formadas (fbf's) de CP1** es generen amb les regles següents:
  1. Tota fòrmula atòmica és una fbf.
  2. Si A i B son fbf's i x és una variable, aleshores són fbf's les fòrmules següents:
     - $\forall x A$
     - $\exists x A$
     - (A)
     - $\neg A$
     - $A \wedge B$
     - $A \vee B$
     - $A \to B$
     - $A \leftrightarrow B$

Altres característiques del llenguatge a analitzar són les següents:

- A nivell lèxic considereu les especificacions de la 1a pràctica.
- Les fòrmules a analitzar estaran en un fitxer de test. Proporcioneu el nom del fitxer d'entrada com millor us vagi depenent del sistema escollit.
- El caràcter nova línia actuarà com a marca de final de fòrmula i la marca de final d'arxiu EOF com a marca de fi d'entrada.
- Desprecieu els caràcters blancs i tabuladors existents a l'entrada.
- Definir un format per a incloure línies de comentaris. L'entrada podrà contenir comentaris.
- Verifiqueu que el llenguatge acceptat per l'analitzador sintàctic és el desitjat. És a dir, verifiqueu que elimineu tots els conflictes mitjançant la definició, per a cada operador lògic i quantificador, el corresponent nivell de precedència i associativitat. La precedència de major (1) a menor (5) prioritat dels operadors és la següent:

  1. $\neg$ (associatiu per la dreta)
  2. $\forall$ i $\exists$ (associatius per la dreta)
  3. $\wedge$ (associatiu per l'esquerra)
  4. $\vee$ (associatiu per l'esquerra)
  5. $\rightarrow$ i $\leftrightarrow$ (associatiu per l'esquerra)

  D'altra banda, els parèntesis permeten modificar la prioritat associada amb els operadors.
- Incorporeu el tractament d'errors, seguint l'estratègia del mode pànic, a nivell sintàctic. Els possibles errors lèxics seran tractats a nivell sintàctic. Indiqueu el número de línia on es detecta l'error.
- Sortida de l'analitzador: Per a cada fòrmula indicar si és o no correcta.

Exercici 5. Càlcul del conjunt de primers dels constructors

En aquest exercici heu d'implementar una eina que calculi el conjunt de primers dels constructors (no terminals) d'una gramàtica LL(1) que no conté cicles ni produccions buides.

El format de la gramàtica incontextual serà el següent:

- Els constructors els especificarem amb lletres majúscules, es a dir, rang `[A-Z]`.
- Els terminals els especificarem amb lletres minúscules, es a dir, rang `[a-z]`.
- El símbol de producció serà els dos punts `:`
- El símbol d'alternativa serà els la barra vertical `|`
- El símbol de final de regla de producció serà el punt i coma `;`

Es demana:

- Implementeu una especificació lèxica i sintàctica que donat un fitxer que conté una gramàtica LL(1) amb el format descrit i que no conté cicles ni produccions buides, calculi el conjunt de primers per a cada constructor.

- Desprecieu els caràcters blancs, tabuladors i new line existents a l'entrada.
- L'entrada podrà contenir comentaris indicats amb el format `//` i acabats en new line. Desprecieu els comentaris a nivell lèxic.
- El final de gramàtica vindrà donat per la marca de final d'arxiu EOF.
- La sortida la podeu escriure directament a la sortida estàndar o a un fitxer.
- Incorporeu el tractament d'errors, seguint l'estratègia del mode pànic, a nivell sintàctic. Els possibles errors lèxics seran tractats a nivell sintàctic. Indiqueu el número de línia on es detecta l'error.
- Per exemple, per calcular el conjunt de primers de cada constructor, podeu utilitzar una estructura de dades estàtica de la forma:

$$\text{Constructors} \times (\text{Terminals} \cup \text{Constructors}) \rightarrow \text{Booleans}$$

Exemple:

```
// Exemple gramàtica LL(1) sense cicles ni produccions buides
S   :   aAc |   fA   |   Bdef
    ;

B   :   Ab  |   Ad   |   k
    ;

A   :   ilm  |  ml
    ;
```

Solució:

```
S= a f i k m
B= i k m
A= i m
```

Exercici 6. Exercici opcional: Lleguatge de descripció d'AF (0,5/1 punt)

1. (0,5 punts) Amplieu la implementació de l'analitzador lèxic per a la descripció d'AF Determinista (AFD) desenvolupat com a exercici optatiu a la 1a pràctica, amb la fase d'anàlisi sintàctica. L'objectiu és construir a memòria la taula de transicions descrita mitjançant l'especificació de l'AFD, minimitzar l'estructura resultant i mostrar el resultat (descripció de l'AFD mínim). Als apunts de l'assignatura, al Tema 2, l'Algorisme 2.4 (Minimització del nombre d'estats d'un AFD) presenta un algorisme per a la minimització d'estats. Prepareu un fitxer README indicant les característiques del llenguatge escollit.
2. (1 punt) En cas que considereu un AF No determinista (AFN), l'objectiu és construir a memòria la taula de transicions descrita mitjançant l'especificació de l'AFN, determinitzar l'estructura resultant i mostrar el resultat (descripció de l'AFN). Als apunts de l'assignatura, al Tema 2, l'Algorisme 2.5 (Determinització d'autòmats finits no deterministes) presenta un algorisme per a la determinització de les transicions. Prepareu un fitxer README indicant les característiques del llenguatge escollit.

Lliurament

La documentació a lliurar per a cada exercici de programació és la següent:

1. Especificació lèxica i sintàctica.
2. Mòduls auxiliars emprats en la implementació de la solució.
3. Joc de proves utilitzat per a la validació de l'exercici.
4. Si ho considereu oportú, un fitxer README amb les particularitats pròpies de la vostra implementació.
5. Important: En cas que la vostra implementació estigui basada en codi de tercers, per exemple disponible a GitHub, ho heu d'indicar al README.

Lliurament de la pràctica al `cv.udl.cat` dins d'activitats. Lliurar un únic arxiu que agrupi tots els exercicis que presenteu, mitjançant la utilitat `tar`. Per a cada exercici: tots els fitxers font, script, joc de proves utilitzat per a la validació i el fitxer README.

Avaluació

- La pràctica la podeu realitzar de manera individual o en grups de 2 o 3 persones.
- El pes d'aquesta pràctica és d'un 20% sobre la nota final de l'assignatura.
- La data límit per lliurar la pràctica és el 25 d'abril per a l'avaluació contínua. Pels que no la tingueu acabada, la podeu lliurar fins el dilluns 29 de maig (data de l'examen del 2n parcial de l'assignatura).