DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 2

Miércoles 29 de julio de 2015 — Teoría de Lenguajes

Integrante	LU	Correo electrónico
Aleman, Damián Eliel	377/10	damian_8591@hotmail.com
Gauna, Claudio Andrés	733/06	gauna_claudio@yahoo.com.ar

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellon I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autonoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Índice

1. Introducción

El objetivo del trabajo practico es implementar un parser para un lenguaje orientado a la composicion de piezas musicales, llamado Musileng, que luego sera transformado al formato MIDI 1 para su reproduccion.
Los pasos que seguimos para realizar el trabajo practico fueron:

- Generar la gramatica adecuada, que sintetice el lenguaje orientado a la composicion de las piezas musicales
- Escribir los terminales del lenguaje y las reglas del lexer
- Escribir los no termianles del lengueje y las reglas del parser.
- Agregarle semantica para que pueda imprimir al lenguaje intermedio que pueda ser leido por el programa midcomp
- Luego de finalizada la traduccion, pueda transformarse a MIDI (.mid) por medio del programa midicomp

Utilizamos para el trabajo la herramienta ANTLR para generar el parser (y el lexer) necesitado.

2. Grámatica

Primero mostramos la gramatica del lexer que lee el archivo como secuencia de caracteres y los mapea a una secuencia de simbolos terminales.
El conjunto de terminales es: {`NUMERAL, TEMPO, DURACION, NUM, COMPAS, SLASH, CONST, NOMBRE, IGUAL, VOZ, LPAREN, RPAREN, LBRACE, RBRACE, REPETIR, SILENCIO, PUNTILLO, ALTURA, OCTAVA, COMA, PUNTOYCOMA` }

A continuación definimos la gramática que generamos para sintetizar el lenguaje: Las terminales de la gramática son todas las cadenas que están entre comillas simples y los que se derivan a partir de una regla (de los tokens en mayúscula).

2.1. Lexer

A la gramatica del lexer, le dimos reglas para que pueda reconocer los tokens:
TEMPO: 'tempo';
COMPAS : 'compas';
CONST: 'const';
IGUAL: '=';
VOZ: 'voz';
LPAREN: '(' ;
RPAREN: ')' ;
LBRACE: '' ;
RBRACE: '' ;
NUMERAL: '#';
COMA: ',' ;
PUNTOYCOMA: ';';
SLASH: '/';
REPETIR: 'repetir';
SILENCIO: 'silencio';
NOTA: 'nota';
PUNTILLO : '.';
DURACION : ('redonda' |'blanca'|'negra'|'corchea'|'semicorchea'|'fusa'|'semifusa');
ALTURA : ('do'| 're'| 'mi'| 'fa'| 'sol'| 'la'| 'si'|'do+'| 're+'| 'mi+'| 'fa+'| 'sol+'| 'la+'| 'si+'|
'do-'| 're-'| 'mi-'| 'fa-'| 'sol-'| 'la-'| 'si-');
NUM: [0-9]+;
NOMBRE: [a-zA-Z_]+;
OCTAVA: [1-9];
COMENTARIOS: $'\\' \sim[\backslash r\backslash$ n]* $\rightarrow$ skip ;
WS : [$\backslash t\backslash r\backslash n$] + $\rightarrow$ skip ;

2.2. Parser

A continuación mostramos la gramatica del parser:
S → tempos elcompas constantes melodia
tempos → NUMERAL TEMPO DURACION NUM
elcompas → NUMERAL COMPAS NUM SLASH NUM
constantes → constante*
constante → CONST NOMBRE IGUAL NUM
melodia → (VOZ LPAREN LPAREN NUM RPAREN { compases})+
| (VOZ LPAREN texto RPAREN { compases})+
compases → compas compases | repeticion compases
repeticion → REPETIR LPAREN NUM RPAREN LBRACE compas RBRACE
compas → (COMPAS LBRACE nota RBRACE)+ | (COMPAS LPAREN silencio RPAREN)+
silencio → SILENCIO LPAREN DURACION PUNTILLO? RPAREN
nota → NOTA LPAREN ALTURA COMA octava, DURACION PUNTILLO? RPAREN
octava → OCTAVA | NOMBRE
El símbolo distinguido es S.

El conjunto de no terminales es:
{`S, tempos, elcompas,contantes, contante, melodia, compases, repeticion, compas, silencio,nota, octava`}

3. Atributos

Asignamos atributos para verificar las restricciones que tenemos que hacer de modo tal que la gramatica genere el lenguaje que necesitamos. Los atributos tambien los usaremos para realizar la traduccion al lenguaje intermedio para que sea legible por el programa midcomp.

Los atributos sintetizados son: {`partitura, tempo, indicacion, listaCompases, voces, repeticiones, compasObj, silencioObj,notaObj, valor`}

Los atributos heredados son: {`indicacion`}
Ahora haremos una breve explicacion de cada atributo:

- partitura: almacena el tempo, la indicacion y la lista de voces.
- tempo: Este atributo guarda la informacion de la duracion de la figura y la cantidad de veces que entra esa figura en un minuto.
- indicacion: Almacena el numerador y el denominador definidos del compas.
- voces:La lista de voces de la melodia
- listaCompases: la lista de los compasObj y las repetcionesObj
- compasObj: Tiene la lista de notas de cada compas
- repeticiones: almacena la cantidad de repeticiones
- notaObj: Almacena la altura,de que octava es, la duracion, la alteracion y si tiene puntillo
- silencioObj: Tiene la lista de silencios de cada compas
- valor: el valor de la octava
- listaCompases: Tiene todos los compases definidos

4. Modo de uso

5. Tests

Realizamos una serie de tests para verificar que el parser funciona correctamente. Para ello le dimos como entrada al parser, archivos invalidos:

- octava 10, es decir fuera del rango del 0 al 9.
- Repetir 0 veces una lista de compases.

- voces:La lista de voces de la melodia
- Distinta duracion de dos compases.
- Instrumento fuera de rango del 1 al 127.
- dos declaraciones distintas de una misma constante.
- Sin voz.

6. Conclusion