E79 Procesadores de lenguaje

Examen de teoría (30 de junio de 2003)

PREGUNTA 1 (6 PUNTOS)

A continuación, se presentan tres posibles extensiones del lenguaje MM3. Elige dos de ellas y explica claramente qué modificaciones se tendrían que hacer en un compilador de MM3 a Stan para que las aceptara. Las modificaciones son independientes entre sí; no hace falta que consideres sus posibles interacciones.

En tu descripción de las modificaciones, procura ser claro, escueto y preciso. En particular, no es necesario que describas partes del compilador que no estén afectadas por las modificaciones. Puedes optar por descripciones algorítmicas o en lenguaje natural para lograr una mayor sencillez en la explicación. También puede facilitarte la exposición una estructura que siga las distintas etapas del compilador.

Explicita cualquier asunción que hagas acerca del compilador o del enunciado propuesto.

Norma de los vectores

Con esta modificación se introduce un nuevo operador unario que permite calcular la norma de un vector, definida como

$$||v|| = \sqrt{\sum_{i=l}^{m} v_i^2},$$

donde v es un vector con índices desde l hasta m. El tipo del resultado es real, con independencia del tipo base del vector. Para utilizar el operador, se escribe el identificador correspondiente al vector encerrado entre dos dobles barras (`||`). No puede haber ninguna separación entre las barras, pero sí entre estas y el vector.

Un ejemplo de uso sería:

```
vector real [1..3] p1;
...
salida <- "El doble de la distancia de p1 al origen es " <- 2*||p1|| <- "$n";
...
```

Puedes asumir que existe la instrucción `FSQRT` que calcula la raíz cuadrada del tope de la pila de reales y deja en ella el resultado.

Nota: el código generado no puede crecer exponencialmente con la talla del programa.

Información sobre vectores

Con esta extensión se añaden atributos a los identificadores de vectores que permiten al programa consultar su tamaño (`talla`) y sus límites inferior (`inf`) y superior (`sup`). Se accede a estos atributos escribiendo, posiblemente separados por espacios o comentarios, un punto y el nombre del atributo detrás del nombre del vector. El valor devuelto es de tipo entero.

Por ejemplo, si queremos escribir el doble de los elementos del vector `v` utilizando esta nueva extensión, podríamos emplear el siguiente bucle:

```
repite v.talla veces:
  salida <- 2*v[v.inf+contador-1] <- "$n";
fin
```

Y para escribirlos en orden inverso:

```
repite v.talla veces:
  salida <- 2*v[v.sup-contador+1] <- "$n";
fin
```

Los nombres de los atributos son nuevas palabras reservadas que siguen las normas habituales de MM3.

Interrupción de bucles

Esta modificación introduce dos nuevas sentencias que permiten terminar o volver al principio de un bucle, de manera similar a las órdenes `break` y `continue` de C. Para ello, se emplean las nuevas palabras reservadas `termina` y `reinicia`. La sentencia `termina;` supone la terminación del bucle en que se encuentra y la continuación de la ejecución por la sentencia inmediatamente siguiente al bucle. En cuanto a la sentencia `reinicia;`, en los bucles `mientras` supone el salto a la evaluación de la condición y en los bucles `repite` supone el incremento del `contador` correspondiente y la nueva ejecución del cuerpo si no se hubiera alcanzado el final.

Un ejemplo de uso sería el siguiente bucle que escribe una tabla con los resultados no primos del cálculo de una función:

```
repite n veces:
  res<- f(contador);
  si esprimo(res):
    reinicia;
  fin
  salida <- contador <- "," <- res <- "$n";
fin
```

El utilizar estas nuevas sentencias fuera de un bucle es un error que debe ser señalado por el compilador.

Nota: se debe explicar someramente cómo se evita que la salida prematura de un bucle `repite` provoque problemas con la representación que se utilice para el `contador` o el límite.

Pregunta 2 (2 puntos)

Escribe expresiones regulares o autómatas finitos deterministas; o bien demuestra que no existen, para los siguientes lenguajes:

- Cadenas de ceros y unos que representan caracteres ASCII imprimibles (7 bits con valor entre 32 y 127).
- Comentarios de un lenguaje de programación que empiezan por la secuencia `{{*`, terminan con `*}}` y no contienen ni saltos de línea, ni ninguna secuencia `*}}`.
- Cadenas formadas únicamente por los tokens **id** (identificador), **entrada** (palabra clave `entrada`), **corcheteAb** (corchete abierto), **corcheteCer** (corchete cerrado), **flecha** (la secuencia `->`), **pyc** (punto y coma) y que son sentencias válidas en MM3.
- Literales de cadena de MM3 con, al menos, una secuencia de escape.

Pregunta 3 (2 puntos)

Sea G una gramática, ya aumentada, con n no terminales, tal que la parte derecha más larga de una regla no tiene más de m símbolos. Demuestra:

- Si G es LR(0), el autómata de prefijos viables tiene al menos $m + n$ estados.
- Si G es SLR, el autómata de prefijos viables tiene al menos máx(m, n) estados.

Nota: el estado generado por el ítem $\langle \mathsf{S}' \rangle \rightarrow \langle \mathsf{S} \rangle \mathbf{\$} \cdot$ se cuenta en el caso del autómata LR(0) pero no para el caso SLR. ($\langle \mathsf{S}' \rangle$ es el no terminal añadido para aumentar la gramática y $\langle \mathsf{S} \rangle$ el no terminal inicial de la gramática original).