Las Etapas de la Compilación

El Parsing en la Compilación

Universidad de Cantabria

Outline

Las Etapas y el Parsing

¿Qué pasos son necesarios dar para hallar la estructura de un programa?

¿En qué pasos podemos dividirlos de forma racional?

Las Etapas y el Parsing

¿Qué pasos son necesarios dar para hallar la estructura de un programa?
¿En qué pasos podemos dividirlos de forma racional?

Las Etapas y el Parsing

Un autómata no necesita leer la cinta de lectura varias veces. Pero puede ser interesante para el humano que el compilador haga varias pasadas al fichero fuente o que lo transforme.

Un ejemplo: Tokens

Los tokens son grupos de caracteres, tal que unidos tienen una significación especial.

Un ejemplo: Tokens

Ejemplos de token:

- Valores numéricos: 4.5
- Strings : “hola”
- Palabras reservadas: if, while, for.

Un ejemplo: Tokens

Los tokens pueden ser diferentes dependiendo del programa, esto se hace por conveniencia, para mantener el código legible y por no poner limitación a acciones.

Análisis Léxico

El análisis léxico se ocupa de evaluar las reglas básicas del código fuente, que cada uno de los **tokens** introducidos sean conocidos.

Análisis Léxico

Para ello, cada uno de los tokens es introducido como expresión regular y convertido a un autómata que detecta si un token coincide con una expresión regular.

Análisis Léxico

Al final, devuelve un flujo (“stream”) de tokens que los utilizará el analizador sintáctico (parser).

Análisis Sintáctico

De nuevo usaremos el problema de palabra como referente y como sustrato las gramáticas libres de contexto y los autómatas con pila.

Análisis Sintáctico

Así como en el análisis léxico, tenemos el autómata mínimo es la solución a seguir, en el análisis sintáctico hay varias técnicas para resolver el problema de la palabra.

Análisis Sintáctico

- Utilizar un algoritmo genérico para hallar una derivación de la palabra.
- Algoritmo basados en propiedades de la gramática:
  - Las derivaciones a la derecha son facilmente adivinables (Algoritmos LR)
  - Las derivaciones a la izquierda son facilmente adivinables (Algoritmos LL)

Análisis Sintáctico

Normalmente se trabajará con autómatas con pila con alguna propiedad adicional, pero es importante **recordar que estos autómatas no tienen una capacidad expresiva mayor.**

Generación de Código Intermedio

En esta fase se genera a partir del código fuente un archivo con el programa en código intermedio, a veces llamado código de tres direcciones, ya que a lo sumo involucran a tres terminos.

Generación de Código Intermedio

- Es más fácil de mejorar el rendimiento del programa.
- Puede ser un punto intermedio para generar código para diferentes plataformas.

Optimización de Código Intermedio

Antes de empezar, ¿que es lo que esperamos?
Una posible respuesta es:
Generar código que se ejecute más velozmente:

- Se utilizara el mismo procedimiento que hemos generado..
- en las partes que no influyan al programa.

Optimización de Código Intermedio

Antes de empezar, ¿que es lo que esperamos?
Una posible respuesta es:
Generar código que se ejecute más velozmente:

- Se utilizara el mismo procedimiento que hemos generado..
- en las partes que no influyan al programa.

Generación de Código Final

En la útlima parte de la compilación se genera código final, adaptado a la máquina.

- No se vuelve a intentar optimizar.
- El mismo código intermedio puede ser utilizado para generar código para otras plataformas.