Proyecto 3 Compiladores

Especificación

En este proyecto los grupos deben adaptar el compilador de triangulo y su ambiente interactivo de Netbeans en su versión simple, sabor a vainilla (solo while, sin for ni ninguna otra estructura de control), para que genere código de LLVM y ejecute nativamente en el procesador de la máquina. Para esto ustedes deben:

1. Agregar una ventana adicional al IDE-Triangle en la cual se desplegará el código generado de LLVM.
2. Ofrecer la opción en el menú de pull-down de compilar a LLVM.
3. No deshabilitar la compilación ni la ejecución de código nativo de TAM. La generación de código a LLVM es una opción.
4. Ofrecer la opción de guardar el código LLVM a disco.
5. Ofrecer la opción de compilar el código LLVM a código nativo de la máquina.
6. Ofrecer la opción de ejecutar el código LLVM desde el ambiente interactivo en una ventana de shell del sistema operativo.

Para lograr esto debe hacer las siguientes cosas.

1. Instalar el LLVM en la computadora en que va a correr el IDE-Triangle. esto usualmente require instalar también el compilador de C clang, el cual viene ya con la opción de generar código LLVM. Instalar, utilizar, y revisar el código intermedio de LLVM les ayudará a descifrar qué es lo que cada instrucción debe generar. Les recomiendo que se familiaricen lo mas que puedan con el código intermedio de LLVM.

2. Para cada instrucción de triángulo les conviene hacer un programa de prueba equivalente en C, que compile la instrucción a LLVM (también lo puede hacer revisando los manuales y libros que están en el tecDigital sobre LLVM). con esto logra entender como es que el LLVM traduce código fuente a código intermedio.

   por ejemplo: si desea saber como declarar variables enteras en LLVM compilaría el código

   ```
   int n;
   int f() {}
   ```

   equivalente a Triangle

   ```
   let
      n : Integer
   in
     begin end
   ```

   ```
   > clang -emit-llvm -S prueba.c -o prueba.ll
   ```

   Debe considerar que el código generado tiene mucho “boiler plate” y que se puede simplificar mucho más. Para esto están los textos como el LLVM Cookbook. Para el caso anterior, el código generado de interés es la declaración del identificador. Que en llvm genera el siguiente código

```
@n = dso_local global i32, align 4
```

Debe tomar en cuenta que en Triangle todo se guarda en la pila, y pueden haber varios niveles anidados de visibilidad (ámbitos) con funciones o procedimientos dentro de otras funciones o procedimientos. El C, por el contrario, solo tiene dos niveles, las variables globales y las variables locales a cada procedimiento/función.

Puede suponer que el primer nivel de delcaración en Triangle corresponde a variables globales, y el segundo a locales a procedimientos. Programas de Triangle que tengan funciones dentro de funciones no se espera que generen código en LLVM, ya que el C no permite esto, sin embargo, es completamente posible hacerlo, pero no podrá buscar el código utilizando el código C que genera el clang, ya que el C no lo permite. Hay otros compiladores que generan código LLVM y que sí tienen esta anidación, puede buscar esos. También puede encontrar cómo se hace esto consultando la documentación de los libros sobre LLVM que están en el tecDigital, se darán 15% extra si lo logra.

Puede también para familiarizarse con LLVM, tratar de generar código directamente como si fuera ensamblador, al fin y al cabo eso es lo que es. Con esto podrá revisar qué partes del código generado por clang son esenciales y cuales no.

3. Agregar información necesaria en la tabla de símbolos del Triangle para que pueda generar el código de LLVM.

4. Debe agregar un visitador a los árboles sintácticos de triángulo que generen código (texto) LLVM.

5. Agregar en LLVM las bibliotecas necesarias para que sus programas puedan ejecutar en el shell del sistema operativo, estas son básicamente:
   1. Lectura e impresión de números enteros.
   2. Lectura e impresión de caracteres.

   Note que estas rutinas las puede escribir en C, y compilar a LLVM con el compilador de clang.

6. Agregar al IDE-Triangle la opción de correr nativamente sus programas, esto corresponde a tomar el código de Triangle, compilarlo a código intermedio de LLVM, usar las herramientas de LLVM para generar el código nativo en su máquina, y ejecutar dicho programa.

7. Se dará 5% puntos adicionales si al código de LLVM que usted genera, se puede configurar con las optimizaciones que el paquete provee. Esto implica:
   1. Agregar un dialogo de optimizaciones al IDE-Triangle.
   2. Generar el código con estas optimizaciones (para esto el LLVM puede hacer optimizaciones de código intermedio a código intermedio).

Fecha de Entrega

10 de Noviembre a las 12pm, que en el tecDigital aparecerá como el 11 de Noviembre a las 00:00am.