**Examen II – Lenguajes y Autómatas.**

29 – Noviembre – 2018

Nombre: Yulisa Judith Onofre González

Matricula: 15480979

1. **Es la fase cuyo objetivo consiste en modificar el código objeto generado.**

Optimización de código

1. **por el generador de código, para mejorar su rendimiento.**

Compilador

1. **Cuáles son los tipos de optimización.**

* Optimizaciones dependientes de la máquina
* Optimizaciones independientes de la máquina

1. **Ejemplos de optimización dependientes de la máquina.**

* Minimización del uso de registros en máquinas en las que no se disponga de un conjunto de registros muy grande. Puede llegar a generarse código que utilice sólo un registro. Esta cuestión se ha analizado anteriormente, en la Sección 6.1.1.
* Uso de instrucciones especiales de la máquina, que supongan una optimización respecto al uso de construcciones más generales, presentes en todos los lenguajes de máquina.
* Reordenación de código: algunas arquitecturas son más eficientes cuando las operaciones se ejecutan en un orden determinado. Modificando el código para sacar provecho de ese orden se puede optimizar el programa objeto.

1. **Ejemplos de optimización independientes de la máquina.**

* Ejecución parcial del código por parte del compilador, en lugar de retrasar su ejecución al programa objeto.
* Eliminación de código que resulta redundante, porque previamente se ha ejecutado un código equivalente.
* Cambio de orden de algunas instrucciones, que puede dar lugar a un código más eficiente.
* Es frecuente que los bucles sean poco eficientes, porque se ejecuten en su cuerpo instrucciones que podrían estar fuera de él, o porque la reiteración inherente al bucle multiplique la ineficiencia causada por el uso de operaciones costosas, cuando podrían utilizarse otras menos costosas y equivalentes.

1. **Explica con tus palabras como funciona el reordenamiento de código.**

Es el que permite que el código fuente se reduzca a su tamaño o que el mismo código objeto sea compilado.

1. **Explica un ejemplo de optimización en tiempo de compilación utilizando cuádruplas.**

Dado que las cuádruplas son una abstracción de los lenguajes ensambladores y de la máquina, podrán usarlo casi todos los compiladores, aunque no generen esta representación intermedia:

* Se supondrá que se dispone de una tabla (T), en la que se conservará la información de las variables del programa fuente: los identificadores y sus valores.
* Se selecciona el conjunto de cuádruplas objeto de la optimización. Casi siempre será el que corresponda a alguna expresión aritmética.
* Se tratan todas las cuádruplas en el orden en el que aparecen y se aplica reiteradas veces el siguiente tratamiento, según su tipo. Para aplicar el tratamiento, hay que utilizar la Tabla 7.1.

|  |  |
| --- | --- |
| Estructura de la cuádrupla | Tratamiento |
| 1. (op, op1, op2, res), op1 es un identificador y (op1, v1) está en T | Se sustituye en la cuádrupla op1 por v1 |
| 1. (op, op1, op2, res), op2 es un identificador y (op2, v) está en la tabla T | Se sustituye en la cuádrupla op2 por v2 |
| 1. op, v1, v2, res), donde v1 y v2 son valores constantes o nulos | Si al evaluar v1 op v2 se produce un error:   * Se avisa del mismo1 y se deja la cuádrupla original.   En otro caso:   * Se elimina la cuádrupla. * Se elimina de T el par (res, v), si existe. * Se añade a T el par (res, v1 op v2). |
| 1. (=, v1, , res) | * Se elimina de T el par (res, v), si existe. * Si v1 es un valor constante, se añade a T el par (res, v1). |

1. **Explica un ejemplo de eliminación de redundancias.**

En este ejemplo, es fácil identificar las redundancias mediante la simple observación del código: en la primera instrucción son necesarias las tres cuádruplas. En la segunda, puesto que b y c no han cambiado de valor (aunque a sí ha cambiado), en lugar de calcular de nuevo el valor de su producto, se puede tomar directamente de la variable auxiliar t1. Para la tercera cuádrupla, no ha cambiado el valor de ninguna de las tres variables, por lo que el valor de la expresión completa puede tomarse directamente de la variable t4.

|  |
| --- |
| (\*, b, c, t1)  (+, a, t1, t2)  (=, t2, , a) |
| (+, a, t1, t4)  (=, t4, , d) |
| (=, t4, , b) |

El objetivo de esta sección es proporcionar un algoritmo que automatice la identificación y reducción de las redundancias.

1. **Explica un ejemplo de reordenamiento de operaciones.**

El reordenamiento de operaciones es cuando se tiene que realizar varias veces el mismo resultado, se tiene que generar antes de que se esté utilizando, ya que con eso se podrá ver la optimización que esta teniendo.

1. **Quien es el Crush famoso de su profesor.**