LENGUAJE Y AUTOMATAS II

UNIDAD 3

15 DE ABRIL DE 2018

IVÁN ALEJANDRO PADILLA ESPARZA | #15480063
JUAN PABLO ROSAS

INTRODUCCIÓN

En este reporte se hablará sobre la optimización, esta consiste en mejorar el código intermedio de tal modo que quede un código maquina más rápido de ejecutar. Hay varios tipos de optimización locales, de ciclos, globales o de mirilla al igual que se observarán los costos, manejando los costos de ejecución sus criterios para mejorar el código o las herramientas para el análisis del flujo de datos.

También se menciona sobre los costos que esto implica, como se puede mejorar el código y unas herramientas para el óptimo entendimiento del flujo de datos.

1. Tipos de optimización

Las optimizaciones pueden realizarse de diferentes formas. Las optimizaciones se realizan en base al alcance ofrecido por el compilador.

La optimización va a depender del lenguaje de programación y es directamente proporcional al tiempo de compilación; es decir, entre más optimización mayor tiempo de compilación.

Como el tiempo de optimización es gran consumidor de tiempo (dado que tiene que recorrer todo el árbol de posibles soluciones para el proceso de optimización) la optimización se deja hasta la fase de prueba final.

1.1 Locales

La optimización local se realiza sobre módulos del programa. En la mayoría de las ocasiones a través de funciones, métodos, procedimientos, clases,

La característica de las optimizaciones locales es que sólo se ven reflejados en dichas secciones.

1.2 Ciclos

La mayoría de las optimizaciones sobre ciclos tratan de encontrar elementos que no deben repetirse en un ciclo.

```
while(a == b)
{
int c = a;
c = 5; …;
}
```

En este caso es mejor pasar el int c =a; fuera del ciclo de ser posible.

El problema de la optimización en ciclos y en general radica es que muy difícil saber el uso exacto de algunas instrucciones. Así que no todo código de proceso puede ser optimizado.

1.3 Globales

La optimización global se da con respecto a todo el código.

Este tipo de optimización es más lenta, pero mejora el desempeño general de todo programa.

Las optimizaciones globales pueden depender de la arquitectura de la máquina.

En algunos casos es mejor mantener variables globales para agilizar los procesos (el proceso de declarar variables y eliminarlas toma su tiempo) pero consume más memoria.

Algunas optimizaciones incluyen utilizar como variables registros del CPU,
utilizar instrucciones en ensamblador.

1.4 De Mirilla

La optimización de mirilla trata de estructurar de manera eficiente el flujo del programa, sobre todo en instrucciones de bifurcación como son las decisiones, ciclos y saltos de rutinas.

La idea es tener los saltos lo más cerca de las llamadas, siendo el salto lo más pequeño posible

2. Costos

Los costos son el factor más importante a tomar en cuenta a la hora de optimizar ya que en ocasiones la mejora obtenida puede verse no reflejada en el programa final, pero si ser perjudicial para el equipo de desarrollo.

La optimización de una pequeña mejora tal vez tenga una pequeña ganancia en tiempo o en espacio, pero sale muy costosa en tiempo en generarla.

2.1 Costo de ejecución. (Memoria, registros, pilas)

Costo de ejecución.
Los costos de ejecución son aquellos que vienen implícitos al ejecutar el programa.

Memoria

La memoria es uno de los recursos más importantes de la computadora y, en consecuencia, la parte del sistema operativo responsable de tratar con este recurso, el gestor de memoria, es un componente básico del mismo.

Registros

Los registros del procesador se emplean para controlar instrucciones en ejecución, manejar direccionamiento de memoria y proporcionar capacidad aritmética.

Tipo de registros:

- Registros de Segmento
- Registros de Apuntador de Instrucciones
- Registros Apuntadores
- Registros de Propósito General
- Registros Índices
- Registros de Banderas

La pila

La aparición de lenguajes con estructura de bloque trajo consigo la necesidad de técnicas de alojamiento en memoria más flexibles, que pudieran adaptarse a las demandas de memoria durante la ejecución del programa.

2.2 Criterios para mejorar el código

La mejor manera de optimizar el código es hacer ver a los programadores que optimicen su código desde el inicio, el problema radica en que el costo podría ser muy grande ya que tendría que codificar más y/o hacer su código más legible.

Los criterios de optimización siempre están definidos por el compilador.

Criterios de optimización

Muchos de estos criterios pueden modificarse con directivas del compilador desde el código o de manera externa.

Este proceso lo realizan algunas herramientas del sistema como los ofuscadores para código móvil y código para dispositivos móviles.

2.3 HERRAMIENTAS PARA EL ANÁLISIS DEL FLUJO DE DATOS

Existen algunas herramientas que permiten el análisis y la correcta optimización del flujo de datos entre las más importantes están:

+ **DEPURADOR**
+ **DESAMBLADOR**
+ **DIAGRAMA DE FLUJO**
+ **DICCIONARIO DE DATOS**

DEPURADOR
Es una aplicación que permite correr otros programas, permitiendo al usuario ejercer cierto control sobre los mismos a medida que los estos se ejecutan, y examinar el estado del sistema (variables, registros, banderas, etc.) en el momento en que se presente algún problema.

El depurador permite detener el programa en:

- Un punto determinado mediante un punto de ruptura.
- Un punto determinado bajo ciertas condiciones mediante un punto de ruptura condicional.
- Un momento determinado cuando se cumplan ciertas condiciones.
- Un momento determinado a petición del usuario.

DESAMBLADOR o DESENSAMBLADOR
Es un programa de computadora que traduce el lenguaje de máquina a lenguaje ensamblador, la operación inversa de la que hace el ensamblador.

Un desensamblador se diferencia de un descompilador, en que está dirigido a un lenguaje de alto nivel en vez de al lenguaje ensamblador.

DIAGRAMA DE FLUJO DE DATOS
Es una herramienta de modelización que permite describir, de un sistema, la transformación de entradas en salidas; el DFD también es conocido con el nombre de Modelo de Procesos de Negocios

DICCIONARIO DE DATOS
El Diccionario de Datos es un listado organizado de todos los elementos de datos que son pertinentes para el sistema, con definiciones precisas y rigurosas que le permite al usuario y al proyectista del sistema tener una misma comprensión de las entradas, de las salidas, y también de cálculos intermedios.

CONCLUSION

Al realizar este reporte, nos damos cuenta que la optimización juega un papel muy importante en todo, ya que siempre se está buscando una manera de que las cosas se hagan más rápido para que funcionen mejor, el objetivo de la optimización es mejorar el código objeto creado para que este sea más rápido de ejecutar y tenga un mayor rendimiento, lo que hace la optimización es que compensa las ineficiencias del lenguaje fuente, es decir que quita lo que no se necesita o está de más en el código.

En este reporte también observamos los costos que son el factor más importante a tomar en cuenta cuando se va a optimizar, en esto se maneja costo de ejecución como son de memoria, registros o pilas, los criterios para mejorar un código y las herramientas necesarias para el análisis del flujo de datos.

CONCEPTOS

Optimización: método para determinar los valores de las variables que intervienen en un proceso o sistemas para que el resultado sea el mejor posible.

Compilador: es un programa informático que traduce un programa que ha sido escrito en un lenguaje de programación a un lenguaje común.

Lenguaje de programación: es un lenguaje formal diseñado para realizar procesos que pueden ser llevados a cabo por maquinas como las computadoras.

Módulos: es una porción de un programa de ordenador. De las varias tareas que debe realizar un programa para cumplir con su función u objetivos, un módulo realizara, comúnmente, una de dichas tareas.

Métodos: es una subrutina cuyo código es definido en una clase y puede pertenecer tanto a una clase, como a un objeto o a una instancia.

Clases: es un modelo que define un conjunto de variables y métodos apropiados para operar con dichos datos el comportamiento.

Bifurcación: hace referencia a la creación de una copia de sí mismo por parte del programa, que entonces actúa como un proceso hijo del proceso originario.

Variable: es un espacio en memoria reservado para almacenar un valor que corresponda a un tipo de dato soportado por el lenguaje de programación.

Tiempo: periodo determinado durante el que se realiza una acción o se desarrolla un acontecimiento.

Memoria: En informática, la memoria es el dispositivo que retiene, memoriza o almacena datos informáticos durante algún período de tiempo.

Registros: Un registro es una memoria de alta velocidad y poca capacidad, integrada en el microprocesador, que permite guardar transitoriamente y acceder a valores muy usados, generalmente en operaciones matemáticas.

Pila: Es un método de estructuración datos usando la forma LIFO, que permite almacenar y recuperar datos.

Registros de Segmento: Se utiliza para alinear en un límite de párrafo o dicho de otra forma codifica la dirección de inicio de cada segmento y su dirección en un registro de segmento supone cuatro bits 0 a su derecha.

Registros de Apuntador de Instrucciones: Contiene el desplazamiento de dirección de la siguiente instrucción que se ejecuta. El IP está asociado con el registro CS en el sentido de que el IP indica la instrucción actual dentro del segmento de código que se está ejecutando actualmente.

Registros Apuntadores: El apuntador de la pila de 16 bits está asociado con el registro SS y proporciona un valor de desplazamiento que se refiere a la palabra actual que está siendo procesada en la pila.

Registros de Propósito General: Pueden guardar tanto datos como direcciones. Son fundamentales en la arquitectura de von Neumann. La mayor parte de las computadoras modernas usa GPR.

Registros Índices: Se utilizan en programación como punteros de direcciones de memoria. El cambio de las direcciones especificadas lo realizaremos mediante direccionamiento indirecto. Con los registros índice logramos realizar con una instrucción lo mismo que antes realizábamos con varias instrucciones.

DEPURADOR: Es un programa usado para probar y depurar los errores de otros programas. El código a ser examinado puede alternativamente estar corriendo en un simulador de conjunto de instrucciones.

DESAMBLADOR: Es un programa de computador que traduce el lenguaje de máquina a lenguaje ensamblador, la operación inversa de la que hace el ensamblador.

DIAGRAMA DE FLUJO: El diagrama de flujo o flujograma o diagrama de actividades es la representación gráfica del algoritmo o proceso.

DICCIONARIO DE DATOS: Un diccionario de datos, o repositorio de metadatos, como lo define el IBM Dictionary of Computing, un repositorio centralizado de información sobre datos tales como significado, relación con otros datos, origen, uso y formato.

BIBLIOGRAFÍA

Jesus Carretero (2006), Sistemas Operativos, 15-Abril-2018, de Arcos Info Sitio Web: http://www.arcos.inf.uc3m.es/~ssoo-va/ssoo-va/libro/pdf/cap04.pdf

María Aguilera y Sergio Gálvez (2010), Gestión de la memoria en tiempo de ejecución, 15-Abril-2018, de Traductores, Compiladores e Intérpretes Sitio web: http://www.lcc.uma.es/~galvez/ftp/tci/tictema8.pdf

Juan Guerrero Martínez y José Francés (2010-2011), Gestión de la memoria en tiempo de ejecución, 15-Abril-2018, de Escolar Técnica Superior de Enginyeria Sitio web: http://ocw.uv.es/ingenieria-y-arquitectura/sistemas-electronicos-para-el-tratamiento-de-la-informacion/seti_materiales/seti_ocw_a1.pdf

REPORTE

En este reporte vemos de inicio que la optimización es un proceso para mejorar la ejecución de un programa, puede ser mejorando los tiempos de ejecución o reducir los recursos necesarios para ejecutarse. Estas mejoras en un programa pequeño tal vez sean de unos cuantos milisegundos de diferencia, pero si hablamos de un sistema mucho mayor si se reducen grandes tamaños de tiempos.

Existen 4 tipos de optimización: locales, ciclos, globales y de mirilla. La ***optimización local*** en la mayoría de las veces se realiza en las funciones, los métodos y los procedimientos, etc. Pero más que nada se usa cuando alguna parte ya está critica porque esta optimización es la más rápida. Los ***ciclos*** buscan elementos que no deberían de repetirse en un ciclo ya que si hay un error aquí este se haría más grade por el ciclo ya que se repetiría muchas veces Los ***ciclos globales*** es la más lenta, ya que se trata de ver todo el código. Las ***optimizaciones de mirilla***, que consisten en acortar las distancias de los saltos en los algoritmos del código. Este lo que hace es recorrer el código buscando combinaciones que pueden ser reemplazadas por otras mucho más rápidas.

Los costos son uno de los factores más importantes que se deben tomar en cuenta al momento de realizar la optimización ya que muchas veces la mejora que se haga puede no reflejarse en el programa final, pero si perjudicaría al equipo.

En cuanto los costó de ejecución ya vienen por defecto al ejecutar el programa, al parecer los criterios para optimizar siempre estarán definidos por el compilador y la mejor manera de optimizar el código es haciendo que los programadores lo optimicen desde el inicio, pero esto puede ser muy costoso por la cantidad de código en cuanto a las herramientas el depurador es un programa que prueba y elimina los errores de otros programas y el desensamblador hace lo inverso que el ensamblador.