INSTITUTO TECNOLÓGICO DE NUEVO LEÓN
INGENIERÍA EN SISTEMAS COMPUTACIONALES

LENGUAJES Y AUTÓMATAS II

UNIDAD 3 OPTIMIZACIÓN

Proyecto 3 Elaboración de un resumen que incluya cada uno de los subtemas de la unidad.

ALUMNA: **Yulisa Judith Onofre González**
N° DE CONTROL: **15480979**

MAESTRO: **Ing. Juan Pablo Rosas Baldazo**

Guadalupe, Nuevo León, al 09 de Noviembre del 2018.

PROYECTO 3 OPTIMIZACIÓN

1. Introducción

En este presente resumen se van a explicar los diferentes tipos de optimización que son las mas relevantes para la realización de programas.

Los objetivos de este trabajo es conocer la optimización que debe de tener los programas y cuál es la ejecución mas eficaz que se tiene al realizar un programa.

2. Capitulo 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.

Algunos editores ofrecen una versión de depuración y otra de entrega o final.

La optimización es un proceso que tiene a minimizar o maximizar alguna variable de rendimiento, generalmente tiempo, espacio, procesador, etc.

Desafortunamente no existen optimizador que hagan un programa más rápido y que ocupe menor espacio.

La optimización se realiza reestructurando el código de tal forma que el nuevo código generado tenga mayores beneficios.

La mayoría de los compiladores tienen una optimización baja, se necesita de compiladores especiales para realmente optimizar el código.

Sección 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, etc.

Las características de las optimizaciones locales es que solo se ven reflejadas en dichas secciones.

La optimización local sirve cuando un bloque de programa o sección es crítico, por ejemplo:

- La E/S, la concurrencia, la rapidez y la confiabilidad de un conjunto de instrucciones.

Como el espacio que soluciones es más pequeño la optimización local es más pequeño la optimización local es más rápida.

Ejemplos:

- Eliminación secuencias nulas.
- Reducción de potencia.
- Reacondicionamiento de operandos.

Sección 1.2: Ciclos

Los ciclos son una de las partes más esenciales en el rendimiento de un programa dado que realizan acciones repetitivas, y si dichas acciones están mal realizadas, el problema se hace N veces más grandes.

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

Por ejemplo:

```
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.

Otros usos de la optimización pueden ser el mejoramiento de consultas en SQL o en aplicaciones remotas (sockets, E/S, etc.)

Sección 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 se mejer mantener variables globales para agilizar los procesos (el proceso de declarar variables y eliminarlas todo su tiempo) pero consume mas memoria.

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

Sección 1.4: De mirilla

La optimización de mirilla trata de estructurar eficientemente 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

Ideas básicas:

- Se recorre el código buscando combinaciones de instrucciones que pueden ser reemplazadas por otras equivalentes más eficientes.
- Se utiliza una ventana de n instrucciones y un conjunto de patrones de transformación (patrón, secuencias, remplazan).
- Las nuevas instrucciones son reconsideradas para las futuras optimizaciones.

Ejemplos:

- Eliminación de cargas innecesarias.
- Reducción de potencia.
- Eliminación de cadenas de satos.

3. Capítulo 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.

Pero en cambio si esa optimización se hace por ejemplo en un ciclo, la mejora obtenida puede ser N veces mayor por lo cual el costo se minimiza y es benéfico la mejora.

Por ejemplo:

- for(int i=0; i < 10000; i++); si la ganancia es de 30 ms 300s

Sección 2.1: Costo de ejecución (memoria, registros, pilas)

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

En algunos programas se tiene un mínimo para ejecutar el programa, por lo que el espacio y la velocidad de los microprocesadores son elementos que se deben optimizar para tener un mercado potencial más amplio.

Las aplicaciones multimedia como los videojuegos tienen un costo de ejecución alto por lo cual la optimización de su desempeño es crítica, la gran mayoría de las veces requieren de procesadores rápidos (tarjetas de video) o de muchas memorias.

Otro tipo de aplicaciones que deben de optimizarse son las aplicaciones para dispositivos móviles.

Los dispositivos móviles tienen más limitados que un dispositivo de cómputo convencional razón por la cual, el mejor uso de memoria y otros recursos de hardware tiene mayor rendimiento.

En algunos casos es preferible tener la lógica del negocio más fuerte en otros dispositivos y hacer uso de arquitecturas descentralizadas como cliente/servidor o P2P.

Sección 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

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 ofuscadores para código móviles y código para dispositivos móviles.

Sección 2.3: Herramientas para el análisis del flujo de datos

Existen algunas herramientas que permiten el análisis de los flujos de datos, entre ellas tenemos los depuradores y desambladores.

La optimización al igual que la programación es un arte y no se ha podido sistematizar del todo.

4. Conclusiones

De esta forma, los tipos de optimización es mejorar el programa que se desea para que los objetivos que nos de tengan un rendimiento mayor. La mayoría de estos tipos vienen a compensar ciertas ineficiencias, ya que el concepto de lenguaje de alto nivel (ensamblador), ya que se viene suprimiendo detalles de la maquina objetos para facilitar la tarea de implementar uno cualquier algoritmo que deseamos realizar.

De igual manera, los costos son los factores de las ejecuciones del programa que hace que sea mínima y sea mas agila, ya que el criterio que debe de tener el código debe de ser optimo para los programadores que lo estén realizando.

5. Conceptos

Análisis: es un estudio profundo de un sujeto, objeto o situación con el fin de conocer sus fundamentos, sus bases y motivos de su surgimiento, creación o causas originarias.

Ciclos: es una serie de etapas que van en secuencia.

Código: es una serie de símbolos que por separado no representan nada, pero al combinarlos pueden generar un lenguaje comprensible solo para aquellos quienes lo entiendan.

Costos: es una variable del sector económico que representa la totalidad del gasto económico de una producción

Depuradores: s un programa usado para probar y depurar (eliminar) los errores de otros programas (el programa "objetivo").

Ejecución: se refiere a la realización o la elaboración de algo, al desempeño de una acción o tarea, o a la puesta en funcionamiento de una cosa.

Ensambladores: se refiere a un tipo de programa informático que se encarga de traducir un fichero fuente escrito en un lenguaje **ensamblador**, a un fichero objeto que contiene código máquina, ejecutable directamente por el microprocesador.

Funciones: actividad particular que realiza una persona o una cosa dentro de un sistema de elementos, personas, relaciones, etc., con un fin determinado.

Métodos: es una palabra que proviene del término griego methodos (“camino” o “vía”) y que se refiere al medio utilizado para llegar a un fin.

Optimización: es la acción y efecto de optimizar. Este verbo hace referencia a buscar la mejor manera de realizar una actividad.

Procedimientos: es el modo de proceder o el método que se implementa para llevar a cabo ciertas cosas, tareas o ejecutar determinadas acciones.

Proceso: conjunto de fases sucesivas de un fenómeno o hecho complejo.

Programa: proyecto o planificación ordenada de las distintas partes o actividades que componen algo que se va a realizar.

Sección: parte con forma generalmente geométrica que junto con otras constituye una cosa material o un conjunto de cosas.

6. Bibliografía o Referencias

Enlaces

Gabriela Fernández (2013, Nov) Blogger.com. [Online]. Available: http://gaferz.blogspot.com/2013/11/tipos-de-optimizacion.html

Noelia López (2013, Nov) Blogger. [Online]. Available: http://noeliy22.blogspot.com/2013/11/tipos-de-optimizacion.html

(2001) M.C Juan Oliveres [Online]. Available: http://dsc.itmorelia.edu.mx/~jcolivares/courses/ps207a/ps2_u7.pdf

PROYECTO 3 OPTIMIZACIÓN

1. **Capítulo 1: Tipos de optimización**
    - **Sección 1.1: Locales**

      La optimización local es la realización de los que se va a llevar a cabo en los módulos de los programas, ya que con todas las características que se dese hacer, para que se vean reflejadas.

    - **Sección 1.2: Ciclos**

      La optimización de los ciclos es el rendimiento que se requiere tener en el programa ya que en este tipo se busca que no se repitan los elementos. Ya que el código de este tipo no puede ser tan óptimo.

    - **Sección 1.3: Globales**

      La optimización global es la mas lenta ya que se encuentra en todo el código que se está realizando, ya que se tienen que englobar todas las declaraciones que se estén haciendo.

    - **Sección 1.4: De mirilla**

      La optimización de mirilla es la que busca que todo el código este llena de combinaciones de instrucciones para que sea más eficiente el cogido.

2. **Capítulo 2: Costos**
    - **Sección 2.1: Costo de ejecución (memoria, registros, pilas)**

      El costo de ejecución es rendimiento que se tiene en los programas que se requieran ejecutar, ya que por eso se necesitan de aplicaciones que ayuden a la optimización par que sean más rápidos.

    - **Sección 2.2: Criterios para mejorar el código**

      Los criterios para mejorar el código es ver que el código que se este haciendo sea eficaz, para el momento que se quiera hacer su ejecución, no tenga problemas y tener que codificar de nuevo el código.

    - **Sección 2.3: Herramientas para el análisis del flujo de datos**

      Las herramientas para el análisis del flujo de datos es que permitan la optimización de los programas que sean ejecutados, ya que se tiene que analizar, ya que con todo esto no se puede sistematizar del todo.