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.

ALUMNO: Aaron Ulises Nuñez Avila

N° DE CONTROL: 15480456

MAESTRO: Ing. Juan Pablo Rosas Balazo

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

1 Introduccion

En este documento se explicaran los diferentes tipos de optimización que existen para la realizacion de programas, además de conocer la optimización que debe tener los programas y cual el es mas eficaz al momento de 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 optimizaron es gran consumidor de tiempo la optimización se deja hasta la fase de prueba final. Algunos editores ofrecen una versión de depuración y otros de entrega final, la depuracion es un proceso que tiene a minimizar o maximizar alguna variable de rendimiento, generalmente tiempo, espacio, procesador etc. desafortunada mete no existe optimizador que aga un programa mas rapido y que ocupe menos espacio.
La optimización se realiza reestructurado el código de tal forma que el nuevo código generado tnga mayores beneficios. La mayoría de los compiladores tiene una optimización baja, se necesita de compiladores especiales para realmente optimizar el código.

Sección 1.1: Locales

La optimización local se realiza sobre módulos de programa, en la mayoría de las ocasiones atraves de funciones métodos procedimiento y clases, la característica de los optimizadores locales es que solo se van reflejando en dichas secciones, la optimización local sirve cuando un bloque de programa o seccion es critico por ejemplo:
la E/S la concurrencia, la rapidez y confiablidad de un conjunto de instrucción

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

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.

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.

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.

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

La optimización es un programa el cual tiene como objetivo ofrecer un mejor rendimiento mayor al optimizar ciertos procesos . La gran mayoría esta creada para compensar algunas ineficiencias que se pueden presentar en la programación de manera que se crearon métodos los cuales son útil para optimizar los procesos lo cual facilita la tarea utilizando algoritmos, ciclos o condiciones las cuales facilitan el proceso.

Los factores son los costos de las ejecuciones del programa los cuales hacen que sean mínima y sean mas ligeras, ya que debe ser óptimo para los programadores que lo están desarrollando

5 Conceptos

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").

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.

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

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

6. Bibliografía o Referencias

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

Diccionario Google
https://www.google.com/search?client=firefox-b-ab&q=Diccionario

Proyecto 3 Optimización

1-Capitulo 1: Tipos de optimización

Seccion1.1:Locales

> La optimización son las actividades que se llevan a acabo en los diferentes módulos que hay en la programación utilizando los diferentes métodos que se pueden implementar para realizar las distintas acciones además de que estas implementaciones solo son reflejadas en la parte donde son aplicadas.

Sección1.2:Ciclos

>La optimización de los ciclos consiste en hacer que los procesos no sean repetitivos sino que estén ciclados en una sola parte del código sin abarcar gran espacio en el código con la dificultad que estos procesos solo pueden ser aplicados a partes del código muy especificas ya que trabaja con datos o procesos muy concretos.

Seccion1.3:Globales

>La optimización Global es de las mas lentas ya que esta es aplicada a todo el código para poder englobar todas las declaraciones que se están realizando pero como desventaja en el proceso esta optimización consume mas memoria.

Seccion1.4:De mirilla

>La optimización de mirilla trata de estructurar el programa en especial en los ciclos además de que se pueden remplazar instrucciones en caso de que exista una mas eficiente.

2-Capitulo 2: Costos

Sección2.1:Costos de ejecución (memoria, registro, pilas)

>El costo de la ejecución es el rendimiento que tienen los programas al momento de ejecución ya que es requerido de otros programas para poder optimizar el proceso o en algunos casos de requiere de tarjetas de vídeo en el caso de los videojuegos.

Sección2.2:Criterios para mejorar el código

>En este caso se recomienda hacer entendible a los programadores que el código tiene que ser optimizado desde su desarrollo, los criterios generalmente están dados por los compiladores y muchos de estos criterios pueden ser modificados de manera externa al compilador.

Sección2.3:Herramientas para el análisis de flujo de los datos

>Las herramientas permiten que el análisis de datos del programa optimice la ejecución o en proceso de la misma tarea.