Tecnológico Nacional de México.

Instituto Tecnológico de Nuevo León.

Ingeniería en Sistemas Computacionales

Lenguajes Autómatas II

"Optimización y Costos"

Trabajo presentado por:

García Hernández Julio César.

#15480089

Guadalupe, Nuevo León a 9 de Noviembre del 2018.

Índice

Introducción

En este proyecto se abarcarán 2 temas muy importantes, que son la optimización y los costos de ejecución. La optimización es un proceso que tiene a minimizar o maximizar alguna variable de rendimiento, generalmente tiempo, espacio, procesador, etc.

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

Capítulo 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, etc. La característica de las optimizaciones locales es que sólo se ven reflejados 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 confiabilidad de un conjunto de instrucciones. Como el espacio de soluciones es más pequeño la optimización local es más rápida.

Entre las diferentes optimizaciones locales encontramos:

- Simplificaciones algebraicas
- Propagación de copias y constantes.
- Constant folding
- Eliminación de subexpresiones redundantes o comunes.
- Eliminación de código incansable

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.

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.

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.

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.

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

Los dispositivos móviles tienen recursos 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.

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 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: 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.

DESAMBLADOR: 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.

Conclusiones

Tanto la optimización como los costos de esta son muy importantes a la hora de realizar un código. Existen diferentes modos de optimización los cuales se adecuan según sea el caso. Todas con el mismo fin, buscar mejorar el rendimiento del código.

La optimización no solo la desarrollan los programas y/o programadores, todos pueden participar dentro de una empresa desde el usuario final hasta los propios ejecutivos. El objetivo de las técnicas de optimización es mejorar el programa objeto para que nos dé un rendimiento mayor. La mayoría de estas técnicas vienen a compensar ciertas ineficiencias que son inherentes al concepto de lenguaje de alto nivel, el cual suprime detalles de la maquina objeto para facilitar la tarea de implementar un algoritmo.

Por otro lado, los costos vienen al ejecutar un programa, hay algunos que ocupan más o menos memoria que otros. Algunas veces realizar una optimización puede traer ventajas comparadas al tiempo, aunque puede generar costos en otras opciones.

Conceptos

Optimización

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.

Costos

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. 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.
Registros de Banderas: De los 16 bits del registro de banderas, nueve son comunes a toda la familia de procesadores 8086, y sirven para indicar el estado actual de la máquina y el resultado del procesamiento.

Bibliografía

- Alfred V, Compiladores principios, técnicas y herramientas. Segunda edición, Aho. Pearson Educación, México, 2008.
- Tipo de optimización, descrita en http://noeliy22.blogspot.mx/2013/11/tipos-de-optimizacion.html?m=1
- M. Storti, Algoritmos y estructuras de datos. Argentina, 2014

Reporte

Optimización

La finalidad de la optimización es la minimización o maximización de variables dentro del rendimiento entre el tiempo y el proceso. Todo esto para generar más beneficios en nuestro código.

Existen varios tipos de optimación, mencionare algunos:

Local: este se realiza en las entadas y salidas buscando partes del programa en donde sea necesaria la optimización siendo así lo más simple posible.

Ciclos: este se realiza la cantidad de veces que sea necesaria, aunque tiene una desventaja y es que no se sabe exactamente en donde se encuentra el problema y por ende no todo el código puede ser optimizado.

Global: este abarca todo el código haciéndolo más lento. Con esta optimización se agilizan los procesos, pero se consume más memoria. La optimización global se da con respecto a todo el código.

Mirilla: su objetico es estructurar de manera eficiente el programa para poder tener saltos lo más pequeños posibles ante las llamadas.

Costos

Los criterios de optimización están descritos por el compilador, estos criterios pueden modificarse desde el código o de manera externa.

Dentro de los programas se tiene que buscar la forma de optimizarlos para tener costos de ejecución más bajos, estos vienen incluidos al ejecutar dicho programa. La encargada de todo esto es la memoria, la cual se considera como una de las partes más importantes del sistema operativo.

La mejor forma de optimizar es desde el inicio de la programación, diciéndole a los programadores que busquen la manera de hacerlo, aunque esto traería un problema en los costos, ya que se tendría que hacer el código más legible lo que conllevaría en algunos casos, la realización de códigos mas extensos.

A la hora de ejecutar un programa se tienen que optimizar dos elementos muy importantes, el espacio y la velocidad de los microprocesadores.