Instituto Tecnológico de Nuevo León

Unidad 3 Optimización

Nombre: Erick Mauricio Medina Hernandez

Matricula: #13480548

Materia: Lenguajes Autómatas II

Maestro: Rosas Baldazo Juan Pablo

Actividad: Elaboración de un resumen que incluya cada uno de los subtemas de la unidad de la unidad.

Viernes 09 de noviembre del 2018

Índice

Introducción

Los tipos de optimización son importantes conocer ya que nos habla un poco de las secciones que tiene que son 2 locales y costos. Cada punto nos hablara de cómo se maneja en locales en ciclos etc. Y el costo nos habla un poco de cómo se maneja en ejecución ya sea memoria, registro, pilas, etc.

Capítulo 1: Tipos de optimización

Sección 1.1: Locales:

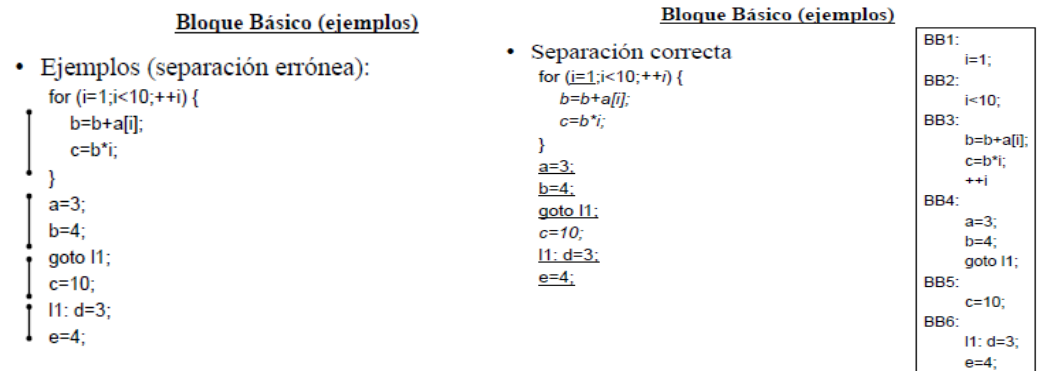
Folding

- Propagación de constantes
- Reducción de potencia
- Reducción de sobrexpresiones comunes

Bloque Básico:

Un bloque básico es un fragmento de código que tiene una única entrada y salida, y cuyas instrucciones se ejecutan secuencialmente. Implicaciones:

- Si se ejecuta una instrucción del bloque se ejecutan todas en un orden conocido en tiempo de compilación.
- La idea del bloque básico es encontrar partes del programa cuyo análisis necesario para la optimización sea lo más simple posible.



Ensamblamiento (Folding). El ensamblamiento es remplazar las expresiones por su resultado cuando se pueden evaluar en tiempo de compilación (resultado constante). Ejemplo: A=2+3+A+C -> A=5+A+C

Estas optimizaciones permiten que el programador utilice cálculos entre constantes representados explícitamente sin introducir ineficiencias.

Implementación del folding durante la generación de código realizada conjuntamente con el análisis sintáctico. Se añade el atributo de constante temporal a los símbolos no terminales y a las variables de la tabla de símbolos. Se añade el procesamiento de las constantes a las reglas de análisis de expresiones. Optimiza: 2+3+b -> 5+b. Hay una suma de constantes (2+3) +b. No optimiza: 2+b+3 -> 2+b+3. No hay una suma de constantes (2+b) +3.

Los bucles son los ciclos de 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.

Los ciclos son 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. Otro uso de la optimización pueden ser el mejoramiento de consultas en SQL o en aplicaciones remotas (sockets, E/S, etc.)

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.

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.

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.

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 del microprocesador 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 (ej. tarjetas de video) o de mucha memoria. El dispositivo móvil tiene 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.

Los 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 buscadores para código móvil y código para dispositivos móviles.

Conceptos

Bloque Básico: Un bloque básico es un fragmento de código que tiene una única entrada y salida, y cuyas instrucciones se ejecutan secuencialmente.

Ensamblamiento (Folding): El ensamblamiento es remplazar las expresiones por su resultado cuando se pueden evaluar en tiempo de compilación (resultado constante).

Implementación del Folding: realizada conjuntamente con el análisis sintáctico. Se añade el atributo de constante temporal a los símbolos no terminales y a las variables de la tabla de símbolos.

Los bucles: Son los ciclos de 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.

Los ciclos: Son 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. Otro uso de la optimización pueden ser el mejoramiento de consultas en SQL o en aplicaciones remotas (sockets, E/S, etc.)

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.

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.

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 del microprocesador son elementos que se deben optimizar para tener un mercado potencial más amplio.

Reporte

Los tipos de optimización, nos habla que tiene dos tipos de optimización que son locales y costos, en lo que son los locales maneja un bloque básico, este hace que un fragmento de código que tiene solo una entrada y salida se ejecutan siguiendo una frecuencia, por ejemplo, si se ejecuta una instrucción en el bloque, automáticamente en un orden se va ejecutando la instrucción marcada si se conoce el tiempo que se va ejecutando. Lo que trata el bloque básico es que hace un análisis en el programa para poder optimizar de una manera simple.

Otra optimización que maneja es la de ensamblamiento, este remplaza los resultados de las expresiones y se puede evaluar el tiempo en que se va compilando. Estas optimizaciones son importantes ya que al programador le sirven para que pueda utilizar cálculos sin introducir ineficiencias.

Una de las optimizaciones lentas que son efectivas, pero tardan más en ejecutarse son las optimizaciones globales ya que es efectiva por que mejora todo el programa a pesar de que es un poco lento en optimizarse, ya solamente dependen de la arquitectura de la máquina para poder optimizarse.

Otra de las optimizaciones en la que se hablan de este tema es la de mirilla y esta funciona bien ya que trata de controlar en un orden de manera en la que se pueda facilitar al momento de ejecutarse.

En el capítulo 2 nos habla un poco de que son los costos y como se manejan, es importante saber que en ocasiones la mejora obtenida puede verse no reflejada en el programa final, 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.

Bibliografía

En Compiladores, Principios, Técnicas y Herramientas, de Sethi, R, Ullman, J Aho A,86-89,94-108,115-131,164-187,190-193,200-201,209-212,221-226,264-274,443-444. U.S.A: Adisson-Wesley Iberoamericana, 1990.

En Compiladores e Intérpretes. Teoría y Practica, de M Moreno, M de la cruz, A Ortega y E Pulido, 78-85, 191-194, 199-204, 221-223. España: Pearson- Prentice Hall, 2006.