MATERIA;
COMPILADORES

DOCENTE:
LUIS GUTIERREZ ALFARO

TAREA: INVESTIGACION

ALUMNO:
CARLOS ANTONIO AGUILAR RAMOS

GRADO Y GRUPO: 6M

LUGAR Y FECHA:
TUXTLA GUTIERREZ CHIAPAS, 05 DE MAYO DE 2023

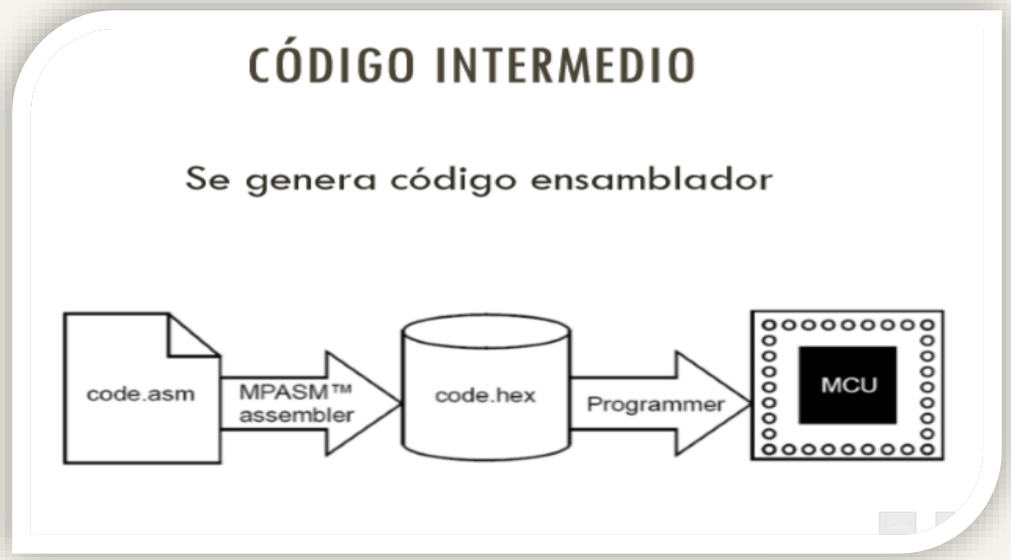
Etapa de optimización de código intermedio de un lenguaje de programación

La optimización de código es el conjunto de fases de un compilador que transforman un fragmento de código en otro fragmento con un comportamiento equivalente y que se ejecuta de forma más eficiente, es decir, usando menos recursos de cálculo como memoria o tiempo de ejecución.

Una optimización es el proceso de transformación de un fragmento de código en otro trozo de código funcionalmente equivalente para mejorar una o varias de sus características. Las dos características más importantes son la velocidad y el tamaño del código. Otras características incluyen la cantidad de energía necesaria para ejecutar el código, el tiempo necesario para compilar el código y, si el código resultante requiere la compilación Just-in-Time (JIT), el tiempo que tarda JIT en compilar el código.

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

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



Tipos de Optimización.

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.

Bucles.

Los bucles 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 bucles 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.

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

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.

Mirillas

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

Bibliografía

https://docplayer.es/63195004-Indice-unidad-2-generacion-de-codigo-intermedio.html

http://dsc.itmorelia.edu.mx/~jcolivares/courses/ps207a/ps2_u7.pdf