Instituto Tecnológico Superior de Jerez

Mapa Conceptual de Manejo de Memoria

Jerez de García Salinas

Fecha

06/09/2018

Alumno:

Mario Alberto Loya Rodríguez

Carrera:

Ingeniería en Sistemas Computacionales

Semestre 4

Materia:

Estructura de Datos

Tema:

1.- Introducción a las Estructuras de Datos

No. de control:

16070135

Profesor:

ISC Salvador Acevedo Sandoval

1. Factores de rendimiento

Importantes factores de influencia para el rendimiento de un programa Java pueden ser separados en dos partes:

- Consumo de memoria del programa Java
- Tiempo de ejecución total de un programa

En caso de que el programa sea en algún caso un programa que interactúa con otros también el tiempo de respuesta es un hecho muy importante de la ejecución de un programa.

Este artículo no cubre la concurrencia. Si desea leer acerca de la simultaneidad / multithreading vea Concurrench Multithreading en Java

Java maneja su memoria en dos áreas. El montón y la pila. Comenzaremos con un breve resumen de memoria en general en una computadora. A continuación, se explica el montón java y la pila.

2.1. Memoria nativa

La memoria nativa es la memoria que está disponible para un proceso, p. en el proceso Java. La memoria nativa es controlada por el sistema operativo (OS) y basada en la memoria física y otros dispositivos físicos: discos, memoria flash, etc.

El procesador (CPU) de la computadora calcula las instrucciones para ejecutar y almacena su computación y resultados en registros. Estos registros son elementos de memoria rápida que almacena el resultado de la CPU. El procesador puede acceder a la memoria normal a través del bus de memoria. Una cantidad de memoria que una CPU puede el acceso se basa en el tamaño de la dirección física que utiliza la CPU para identificar la memoria física. La dirección de 16 bits puede acceder a las ubicaciones de memoria 2^16 (= 65.536). Una dirección de 32 bits puede tener acceso a 2^32 (= 4.294.967.296). Si cada área de memoria consta de 8 bytes, entonces un sistema de 16 bits puede el acceso 64KB de la memoria y el sistema 32-bit puede tener acceso a 468 de memoria.

Un sistema operativo (SO) normalmente utiliza la memoria virtual para asignar la memoria física a la memoria que cada proceso puede ver. El sistema operativo asigna entonces memoria a cada proceso en un espacio de memoria virtual para este proceso y asigna el acceso a esta memoria virtual a la memoria física real.

Los actuales sistemas de 32 bits utilizan una extensión (Extensión de Dirección Física (PAE)) de espacio físico a 36 bits del sistema operativo. Esto permite que el sistema operativo acceda a 64GB. El SO utiliza entonces memoria virtual para

permitir que el proceso individual 4 GB de memoria. Incluso con PAE habilitado, un proceso puede no acceder a más de 4 GB de memoria.

Por supuesto, con un sistema operativo de 64 bits esta limitación 463 ya no existe.

2.2. Memoria en Java

Java gestiona la memoria para su uso. Nuevos objetos creados y colocados en el montón. Una vez que su solicitud ya no tiene referencia alguna a un objeto que el recolector de basura java puede eliminar este objeto y quite la memoria para que su aplicación pueda utilizar esta memoria de nuevo.

2.3.Java Heap

En el montón, la máquina virtual java (IVM) almacena todos los objetos creados por la aplicación java. Para utilizarlo el operador "nuevo". El recolector de basura Java puede separar lógicamente el montón en diferentes áreas, de modo que el pueda identificar más rápidamente los objetos que pueden conseguir quitados.

La memoria para nuevos objetos se asigna en el montón en ejecución hora. Las variables de instancia viven dentro del objeto en el que están declarados.

2.4.Java Stack

La pila es donde se almacenan las invocaciones de método y las variables locales. Si se llama a un método entonces el marco de pila se coloca en la parte superior de la pila de llamadas. El marco de pila mantiene el estado del método incluyendo qué línea de código se está ejecutando y los valores de todas las variables locales. El método en la parte superior de la pila siempre es el método de ejecución actual para esa pila. Los hilos tienen su propia pila de llamadas.

3. Recolector de basura

La JVM recupera automáticamente la memoria que ya no se utiliza. La memoria de los objetos que no se envían más será automáticamente liberados por el recolector de basura. Para ver eso el recolector de basura comienza a trabajar agrega el argumento de la línea de comandos a tu máquina virtual.

Un artículo detallado sobre el recolector de basura se puede encontrar aquí: Tuning Garbage Collection with the máquina virtual Java.

4. Ajustes de memoria para la máquina virtual Java

La JVM se ejecuta con memoria disponible fija. Una vez superada esta memoria, recibirá "java.lang.0ut0 fl \ / IemoryError". El IVM intenta hacer una elección inteligente sobre ellos (consulte la configuración de Java para obtener detalles) pero puede sobrescribir el valor predeterminado con lo siguiente Configuración.

Para dar vuelta al funcionamiento usted puede utilizar ciertos parámetros en la JVM.

Mapa Conceptual de Factores de Rendimiento