Projeto Lógico JVM

Carlos Joel
Gabriel Ferreira Silva
Gabriel Mesquita
Leandro Bergmann
Renato Rangel

Estrutura Básica da JVM

- 1. Carregador de classe (Class Loader)
- 2. Memória (Runtime Data Area)
- 3. Interpretador (Execution Engine)

Carregador de Classe

- Carrega

  Uma classe obtida de um arquivo .class
- Liga
  - Verifica → Se é um arquivo correto .class (java e jvm)
  - Prepara → As estruturas de dados.
  - Resolve → Ref simbólica da Constant Pool para Ref direta.
- Inicializa → Variáveis da classe, fields estáticos e etc.

Memória

- Área de Métodos (Method Area)
- Heap → Malloc e Calloc
- Pilha da JVM (JVM Stack)
- Registro de PC (Pc Register)
- Pilha de Métodos Nativos → Não faremos!

- * Pilha da JVM e registro de PC são um por thread. Como não implementaremos threads, serão somente um em nossa JVM.

Registro de PC

- Tem o endereço da próxima instrução a ser executada.
  - No caso um indice para o array de bytecode (que está dentro da struct code_attribute)

- Como inicializar o PC?
  - Colocar ele na primeira instrução do programa a ser executado pela JVM

Registro de PC

- Como calcular o próximo valor do registro de PC?
  - Caso 1: A instrução é a próxima do método corrente
  - Caso 2: A instrução está no método corrente, mas não é a próxima (ex: goto)
  - Caso 3: A instrução chama um método
  - Caso 4: A instrução retorna para um método

Registro de PC

- Caso 1: Como calcular o próximo valor do registro PC quando a próxima instrução está no mesmo método e é a que é mostrada logo a seguir no bytecode do código?
- Calculável sabendo o número de operandos da instrução atual

Registro de PC

- Caso 2: Como calcular o próximo valor do registro de PC quando a instrução está no frame corrente mas não é a mostrada imediatamente a seguir no bytecode do método (ex: goto)?
- Calculável sabendo o endereço atual e o desvio relativo

Registro de PC

- Caso 3: Como calcular o próximo valor do registro de PC quando a instrução atual invoca um novo método?
- O registro de PC tem então o valor da primeira instrução do novo método.

Registro de PC

- Caso 4: Como calcular o próximo valor de PC quando se está na última instrução de um método e deve-se retornar para outro método?
  - Toda vez que chamamos um método antes de passar para o método chamado salva-se no frame do método chamador o endereço da próxima instrução. Dessa forma, quando voltamos ao método pode-se colocar o valor de PC como sendo igual ao valor salvo.

Pilha da JVM

- ED a ser utilizada: pilha
- Trata-se de uma pilha de frames, onde cada frame é relativo a um método e contém array de variáveis locais, pilha de operandos e uma referência para a constant_pool
- Como desempilhar os frames de tamanho variável?

Pilha da JVM

- Abordagem 1: Cada frame possui apenas uma referência ao array de variáveis locais, uma referência a pilha de operandos e uma referência a constant pool.
- Frame tem tamanho fixo, facilita desalocar da pilha da JVM
- O tamanho do array de variáveis locais e da pilha de operandos é conhecido em tempo de compilação. Desalocar o espaço ocupado por eles também deve ser fácil

Pilha da JVM

- Abordagem 2: Cada frame possui um array de variáveis locais, uma pilha de operandos e uma referência a constant pool.
- Frame tem tamanho variável e é possível desalocar o frame ao conhecer o seu tamanho previamente.

Pilha da JVM

- Como saber quando desalocar um frame?
- Antes de executar um método empilha o frame correspondente e executa. Ao terminar a execução do método desempilha o frame para que a execução prossiga.

Área de Métodos

- Contém informação sobre os métodos e os fields, além da run-time constant pool
- Abordagem 1: representar usando uma lista encadeada, em que cada elemento da lista guardaria os métodos, os fields e a run-time constant pool de uma classe particular.

Área de Métodos

- Pela abordagem 1, para determinar se uma classe está carregada ou não basta iterar pela lista de métodos, checando se o elemento atual corresponde a classe que estamos querendo saber se está carregada ou não

Área de Métodos

- Mas como determinar se um elemento da lista de métodos corresponde a uma determinada classe?
- A partir do atributo this_class do elemento atual descobrimos o nome da classe. Em seguida, comparamos o nome da classe representada pelo elemento com o nome da classe que desejamos saber se está carregada ou não

Heap

- Área onde os objetos criados ficam
- Objetos alocados via malloc e calloc
- ED a ser utilizada: lista
- Cada elemento é uma referência a uma estrutura que contém:
  - Uma referência a informações da classe do objeto
  - Uma referência a uma estrutura que contém os valores das variáveis de instância do objeto

Carregador

- Sabemos como verificar se uma classe foi carregada ou não na memória
- Mas como carregar uma classe na memória?
- Basta ler o .class e salvar as informações na área de métodos.
  - Como já fizemos o leitor, esse passo deve ser fácil

Máquina de Execução

- JIT – Just in Time Compiler ?

- Interpretador ?

Interpretador

- Um interpretador lê, interpreta e executa instruções em bytecode uma a uma. Uma vantagem é que pode rapidamente interpretar um bytecode mas a desvantagem é que executa de forma lenta o resultado interpretado.
  - Não é problema para nós, já que não estamos preocupados com performance

Interpretador

- 1. Lê instrução (fetch)
- 2. Interpreta Instrução
- 3. Executa instrução (execute)
- 4. Vai para a próxima instrução

Interpretador

- Como saber qual a próxima instrução?
- Pelo valor no registro de PC

Fluxo de Execução

- Carrega o .class (ClassLoader)
- Cria o frames correspondente ao método da classe carregada que será usado e empilha na JVM stack.
- Executa o método <init> seguido do método main da classe.
- Para executar acessa o método na method area, inicializa o PC com a primeira instrução, lê, interpreta, e executa. Após executar atualiza o valor do PC. (Loop)
- Percorre a JVM stack executando os métodos e suas instruções.

Observação

- Como fazer quando for executar código de Object.class ou de PrintStream.class? Afinal onde eles ficam no computador varia...
- Colocar eles em uma pasta de .class e acessar a partir dessa pasta