Geração de Rastros de Execução para Visualização de Programas Java

Gabriela Jacques da Silva, Benhur de Oliveira Stein

Universidade Federal de Santa Maria
Laboratório de Sistemas de Computação
{gabijs,benhur}@inf.ufsm.br

Introdução

Java é uma linguagem de programação que vem sendo cada vez mais utilizada para o desenvolvimento de aplicações paralelas e distribuídas, provavelmente porque ela dispõe de mecanismos como invocação de métodos remotos e *threads* que simplificam esse tipo de programação. Apesar de geralmente ser associada com falta de eficiência, esta é uma limitação da implementação da linguagem, e não da linguagem em si [PAN 2001].

Apesar de Java possuir mecanismos que facilitem o desenvolvimento de aplicações distribuídas, este tipo de programação é mais difícil que a programação seqüencial, sendo conveniente a utilização de ferramentas de depuração específicas para este tipo de programação, como ferramentas de visualização. Tais ferramentas necessitam que os programas gerem, durante sua execução, informações que sejam visualizáveis, chamadas rastros de execução.

Este trabalho descreve um agente de geração de rastros em desenvolvimento no Laboratório de Sistemas de Computação, que permitirá a visualização de programas multiprogramados e distribuídos desenvolvidos na linguagem de programação Java. As seções seguintes expõem algumas ferramentas existentes para o monitoramento de programas Java, o agente de rastros, a ferramenta de visualização que está sendo utilizada, algumas conclusões e trabalhos que estão em desenvolvimento.

Monitoramento de Programas Java

Atualmente, existem várias ferramentas de monitoramento da execução de programas Java para a extração de estatísticas de desempenho e visualização. Porém, a maioria delas não pode ser usada de forma adequada para a visualização e depuração de desempenho de aplicações Java distribuídas [OTT 2001].

Um agente de monitoramento de programas Java que já vem agregado ao JDK (*Java Development Kit*) é o HPROF [LIA 1999]. Esta ferramenta coleta e grava informações sobre a execução de um programa Java. O problema de utilizar o HPROF para monitorar programas paralelos é que este não prevê suporte a aplicações distribuídas, ou seja, não faz o monitoramento de comunicações. Desenvolvida pela IBM, Jinsight [GUI 2000] já abrange a visualização de programas. Sua desvantagem é a realização da instrumentação na própria JVM (*Java Virtual Machine*), obrigando o desenvolvedor a ter uma JVM modificada para possibilitar a depuração.

TAU (*Tuning and Analysis Utilities*) é um ambiente de análise de desempenho para programas paralelos complexos que oferece ferramentas de instrumentação, medição e análise [SHE 2001]. Este ambiente foi estendido para permitir a instrumentação de programas Java. Sua vantagem é a possibilidade de selecionar as classes que serão monitoradas, porém a ferramenta de visualização utilizada é VAMPIR [VAM 2002], que é uma ferramenta paga.

As próximas seções descrevem como está sendo implementado o agente de geração de rastros proposto e quais as vantagens em relação as ferramentas expostas acima.

Agente de Geração de Rastros

Para que seja possível a visualização de programas é necessário que estes gerem eventos que representem uma determinada ação. A cada evento está associado um tipo e uma data, além de outros dados dependendo do tipo do evento.

Um dos objetivos deste trabalho é o desenvolvimento de um agente de geração de rastros que possibilita ao programador Java a visualização de suas aplicações sem que seja necessário a modificação no programa em desenvolvimento e também sem a instrumentação da JVM em utilização com chamadas de geração de rastros. Para isso, duas ferramentas estão sendo utilizadas: JVMPI (*Java Virtual Machine Profiler Interface*) [LIA 1999] e uma biblioteca genérica de geração de rastros [JAC 2002], já desenvolvida no Laboratório de Sistemas de Computação.

JVMPI

A JVMPI é uma interface entre a máquina virtual Java e um agente de perfilamento (*profiling agent*). O agente implementa funções que a máquina virtual irá chamar para notificar os eventos ocorridos, enquanto que a máquina virtual disponibiliza funções que permitem seu controle pelo agente, como a seleção dos eventos que serão notificados e a habilitação do coletor de lixo.

Através do uso da JVMPI é possível obter informações sobre todos os eventos ocorridos durante a execução de um programa Java. Seu uso possibilita que os rastros sejam gerados em qualquer máquina virtual que suporte esta interface. Outra vantagem é a possibilidade de monitorar programas sem que seja necessário a instrumentação da própria máquina virtual. Apesar desta abordagem apresentar um custo um pouco maior do que a instrumentação direta, a maioria dos eventos são notificados em situações onde é tolerável a adição do custo de uma chamada de função [LIA 1999].

Biblioteca Genérica de Geração de Rastros

Para a implementação do agente de rastros está sendo utilizada uma biblioteca de geração de rastros [JAC 2002], que registra eventos sem considerar a semântica destes. Esta biblioteca fornece funcionalidades de sincronização de tempo, o que será indispensável para a posterior visualização de programas distribuídos. Outro recurso que está sendo bastante explorado é o suporte da biblioteca à multiprogramação, para o monitoramento de programas Java que façam o uso de múltiplas *threads*. A biblioteca também

ficará responsável pelo gerenciamento dos *buffers* de rastros e pela garantia da atomicidade dos eventos registrados.

As funções de geração de rastros da biblioteca são invocadas sempre que o agente de rastros identifica o evento notificado pela máquina virtual Java como rastreável. O agente toma tal decisão baseado no arquivo de configuração.

Visualização de Programas

A ferramenta de visualização que está sendo utilizada para a análise de comportamento de programas desenvolvidos em Java é Pajé [STE 1999], que é desenvolvida no Laboratório de Sistemas de Computação. Esta ferramenta é adequada, pois é genérica e escalável. Sua genericidade permite que os programas possam ser visualizados de várias maneiras, adequando-se facilmente aos dados que são registrados em uma determinada execução. Sua escalabilidade possibilita a visualização de várias *threads* e várias máquinas virtuais Java em execuções concorrentes, sendo possível visualizar adequadamente uma aplicação paralela.

Conversor de Rastros

Para que os rastros gerados pelo agente sejam vistos em Pajé, é necessário que haja uma ferramenta de conversão de rastros, que tranforme-os em formato Pajé. Esta ferramenta intermediária tem o conhecimento da semântica dos eventos registrados, e assim é capaz de realizar essa conversão.

Atualmente, os eventos que estão sendo monitorados são os de início e fim da máquina virtual, carregamento de classe, inicialização e finalização de *threads*, entrada e saída de métodos, controle de monitores e coleta de lixo.

O agente de rastros ainda permite a configuração, através de um arquivo, das classes e métodos que se deseja monitorar. Isso se faz necessário pelo fato de que a máquina virtual Java notifica todos os eventos para a JVMPI, ou seja, chamadas a métodos de classes internas à máquina virtual são também notificadas. A visualização destes métodos e classes que não são do conhecimento do usuário atrapalhariam a visualização, além de que a geração de rastros para todas as classes e métodos seria bastante intrusiva, podendo alterar o comportamento do programa.

A figura 1 mostra a visualização de um programa Java, onde o fluxo principal faz o lançamento de duas novas *threads*. Estas lançam mais duas *threads* e solicitam a sincronização. O fluxo principal do programa é representado pela primeira régua, enquanto que o lançamento e a sincronização dos outros fluxos são representados pelas flechas. A execução de métodos pode ser visulizada pelos retângulos aninhados.

Conclusões e Trabalhos em Desenvolvimento

O agente de rastros em desenvolvimento atualmente está monitorando programas Java multiprogramados, permitindo a visualização destes na ferramenta Pajé de forma simples e transparente ao desenvolvedor. Assim, já é possível analisar programas Java com a finalidade da melhoria de desempenho.

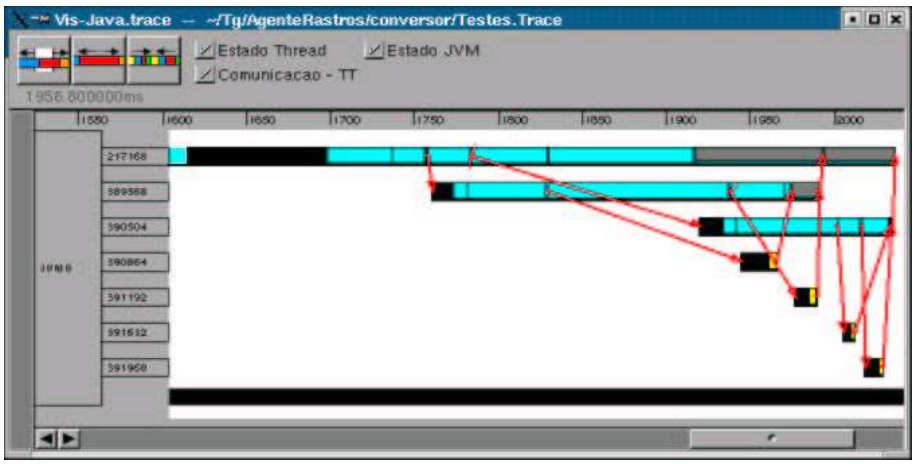

Figura 1: Visualização em Pajé

Para permitir a análise de programas distribuídos está em estudo o monitoramento de invocação remota de métodos. Trabalhos futuros permitirão a visualização de programas paralelos e distribuídos usando a tecnologia de *sockets* disponível na linguagem Java.

Referências

[GUI 2000] GUITART, J. et al. **Instrumentation Environment for Java Threaded Applications.** XI Jornadas de Paralelismo. Granada, Espanha. 2000.

[JAC 2002] JACQUES-SILVA, G.; STEIN, B. O. **Uma Biblioteca Genérica de Geração de Rastros de Execução para Visualização de Programas**. Anais do I Simpósio de Informática da Região Centro do Rio Grande do Sul. Santa Maria. 2002.

[LIA 1999] LIANG, S.; VISWANATHAN, D. **Comprehensive Profiling Support in the Java Virtual Machine**. 5^{th} USENIX Conference on Object-Oriented Technologies and Systems. San Diego, EUA. 1999.

[OTT 2001] OTTOGALLI, F. G. et al. **Visualization of Distributed Applications for Performance Debugging**. International Conference in Computational Science, Lecture Notes in Computer Science 2074, pp.831-840. San Francisco, EUA. 2001

[VAM 2002] Pallas GmbH. **VAMPIR: Visualization and Analysis of MPI Resources**. `http://www.pallas.com/e/products/vampir`

[PAN 2001] PANCAKE, C. M.; LENGAUER, C. **High-Performance Java.** Communications of the ACM, v.44, n.10, p.98–101. Oct. 2001.

[SHE 2001] SHENDE, S.; MALONY, A. D. **Integration and Application of the TAU Performance System in Parallel Java Environments.** Proceedings of the Joint ACM Java Grande - ISCOPE 2001 Conference. Stanford, EUA. 2001.

[STE 1999] STEIN, B. **Visualisation interactive et extensible de programmes parallèles à base de processus légers.** Thèse de doctorat en informatique. Universitè Joseph Fourier, France. 1999.