Cenário atual da Paralelização da ferramenta PajeNG

Jonas Henrique M. Korndorfer, Lucas M. Schnorr

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{jhmkorndorfer,schnorr}@inf.ufrgs.br

Resumo. *Este trabalho apresenta o andamento das otimizações no conjunto de ferramentas para análise de desempenho, PajeNG. Esta simula o comportamento da execução de aplicações paralelas e tem baixo desempenho quando processando grande volume de dados. Neste trabalho são abordados aspectos sobre os resultados das estratégias desenvolvidas para paralelização de partes da ferramenta e o novo modelo em desenvolvimento.*

1. Introdução

Técnicas para análise de desempenho pretendem ajudar o analista a otimizar o programa visando encontrar regiões com baixa utilização dos recursos. Dentre as diversas formas de análise, a técnica de observação comportamental da execução de programas é muito importante, podendo indicar exatamente quanto tempo uma seção de código leva para executar. Frequentemente, a observação comportamental é implementada com base em grandes rastros de execução, gerando visualizações. Assim, sua implementação exige um alto desempenho. Algumas ferramentas implementam esta técnica, como PajeNG [Schnorr 2012] e Vampir [Nagel et al. 1996].

A ferramenta PajeNG[1] contempla um conjunto de ferramentas, de código aberto, para análise de desempenho de aplicações paralelas. Ela permite a simulação do comportamento da execução de aplicações e a exportação dos resultados por visualizações ou por arquivos no formato CSV. Além disso, utiliza um formato de arquivo de rastros próprio [Schnorr 2014], o qual é formado por eventos genéricos sem semântica associada. A vantagem do PajeNG está associada a seu formato de rastro de entrada: seu dinamismo permite a simulação da execução de diversas aplicações que usam diferentes tecnologias para programação paralela. Por outro lado, um dos problemas enfrentados pela ferramenta é o desempenho, devido a sua implementação sequencial dispendendo muito tempo para processar arquivos de rastreamento considerados grandes. Por exemplo, rastros com apenas 700MB já precisam de aproximadamente 30 segundos para serem processados.

Este trabalho faz uma reflexão a partir dos resultados de estratégias desenvolvidas para paralelizar partes do processamento realizado pelo PajeNG. Além disso, apresenta o fluxo de execução que pretendemos alcançar com a nova estratégia em desenvolvimento, visando aumentar o paralelismo e o desempenho.

2. Cenário atual do PajeNG

O fluxo de execução da ferramenta PajeNG pode ser dividido em três etapas. A primeira etapa é responsável por ler o arquivo de rastreamento. A segunda, divide o arquivo em linhas transformando-as em objetos, com base nas definições contidas nos rastros. Por fim,

[1]Disponível em GitHub: https://github.com/schnorr/pajeng

a terceira faz a simulação da execução, armazenando o efeito de cada evento simulado na sua entidade correspondente. Tais objetos são identificados como *container*, que podem ser qualquer entidade monitorável, como processos, máquinas, e *threads*.

Foram desenvolvidas três estratégias para otimizar a ferramenta PajeNG. Estas focaram em agilizar a terceira etapa do fluxo, a simulação. Contudo, todas apresentaram baixo desempenho, mesmo através do processamento desta parte em paralelo. Isto ocorreu porque os demais componentes do fluxo continuam executando sequencialmente, funcionando como um gargalo, limitando o ganho de desempenho.

3. Estratégia de Mapeamento de Múltiplos Fluxos

A etapa seguinte do projeto consiste em executar diversos fluxos em paralelo. Na Figura 1 está representado o modelo a ser desenvolvido, que propõe duas modificações principais. A primeira refere-se a divisão do arquivo de rastros em várias partes, permitindo, assim, o processamento de cada parte em paralelo. A segunda introduz um novo tipo de evento, *PajeTraceFile*, o qual indica a criação de um novo fluxo e a localização de outra parte do arquivo. Os fluxos apresentados na Figura 1 são semelhantes a execução normal do PajeNG, descrito na seção 2, replicados para cada evento *PajeTraceFile* encontrado nos rastros.

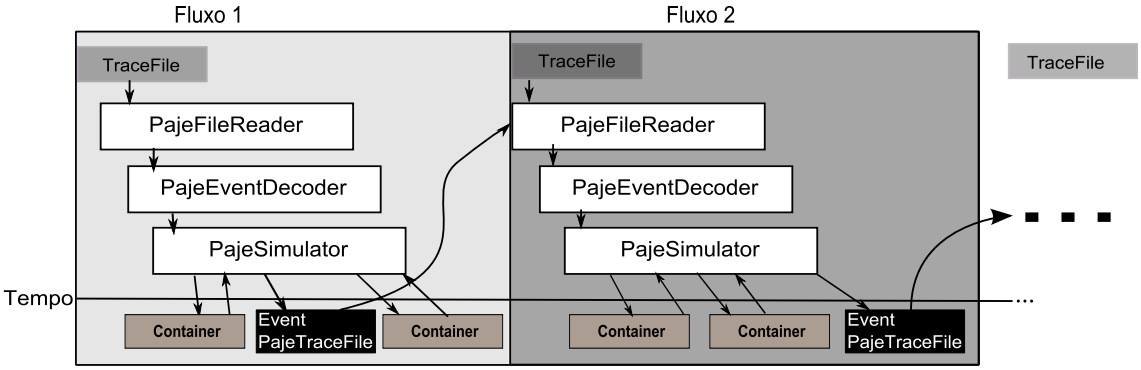

Figura 1. Múltiplos arquivos de rastro e fluxos de execução.

4. Conclusão

Este trabalho está em uma fase exploratória. As primeiras tentativas de otimização do PajeNG mostram que é necessária uma paralelização não trivial do fluxo de execução do programa. Com o modelo proposto na Seção 3, esperamos alcançar um ganho sublinear de desempenho. Isto ocorre porque mesmo com vários fluxos separados, ainda há concorrência na leitura dos arquivos em um único disco.

Referências

Nagel, W. E., Arnold, A., Weber, M., Hoppe, H.-C., and Solchenbach, K. (1996). Vampir: Visualization and analysis of mpi resources.

Schnorr, L. M. (2012). Pajeng – paje next generation. `http://github.com/schnorr/pajeng`.

Schnorr, L. M. (2014). Pajé trace file format. Technical report, UFRGS, Porto Alegre, Brazil. `http://paje.sf.net`.