Mecanismo de Detecção de Tarefas Anômalas Para a Análise de Desempenho de Aplicações com Tarefas Irregulares

Marcelo Cogo Miletto, Lucas Mello Schnorr

Programa de Pós-Graduação em Computação (PPGC/UFRGS), Porto Alegre, Brasil

{marcelo.miletto, schnorr}@inf.ufrgs.br

Resumo. *Este trabalho propõe uma metodologia para enriquecer a análise de desempenho de aplicações baseadas em tarefas, principalmente em cenários onde o custo das tarefas computacionais não é uniforme. Consideramos o emprego de regressão estatística para construir um mecanismo que detecta tarefas anômalas tomando por base o custo estimado por um modelo de desempenho.*

Contextualização e Motivação: Aplicações computacionais provenientes de diversas áreas de pesquisa necessitam de cada vez mais poder computacional. Essa demanda é atualmente suportada pela constante evolução de componentes de hardware voltados para o processamento paralelo e de alto desempenho. No entanto, o desafio de desenvolver aplicações que utilizem tais recursos de forma eficiente torna-se cada vez maior. Uma abordagem bastante popular ultimamente é a programação baseada em tarefas. Tal paradigma permite representar uma aplicação como de um grafo acíclico dirigido, onde os nós são as tarefas computacionais e as arestas suas dependências. Esta ideia, embora seja simples, é bastante poderosa. Ela oferece uma maneira portável de se obter desempenho de aplicações complexas em arquiteturas também complexas [Dongarra et al. 2017]. Este desempenho é obtido pelo do emprego de um sistema de *runtime*. Nele se encontra o escalonador, que tem um papel central em tais aplicações pois é responsável pela distribuição das tarefas entre os recursos computacionais. Para fazer isto de forma eficiente, o escalonador deve considerar aspectos como custos de comunicação e reutilização de cache, enquanto distribui a carga uniformemente entre as unidades computacionais. Diversos trabalhos mostram como esse nível de runtime pode afetar o desempenho de uma mesma aplicação [Miletto and Schnorr 2019]. Isto é causado por sobrecargas no sistema de *runtime* ou por decisões ruins tomadas pelo escalonador. Uma das formas de destacar possíveis problemas de desempenho tanto a nível de *runtime* quanto de aplicação, é pela da análise de tarefas anômalas, que apresentam uma duração maior do que a esperada.

Proposta de Trabalho e Metodologia: Este trabalho propõe uma metodologia para a identificação automática de tarefas anômalas em casos onde a aplicação gera tarefas não homogêneas, como por exemplo, em computações envolvendo matrizes esparsas. O objetivo deste trabalho compreende a elaboração deste mecanismo de detecção, e sua integração em uma ferramenta voltada para a visualização e análise de desempenho de aplicações baseadas em tarefas, chamada StarVZ [Garcia Pinto et al. 2018]. Essa integração é feita como uma forma de validação de nossa metodologia proposta, ao mesmo tempo em que permite explorar casos interessantes para aplicações baseadas em StarPU [Augonnet et al. 2011]. O mecanismo de detecção é baseado em um modelo de regressão, nele, considera-se o número teórico de operações de ponto flutuante que uma tarefa realiza para estimar sua duração. O modelo utilizado define y_i como sendo explicado pela variável $x_i^{2/3}$, onde y_i é o tempo de duração esperado para a tarefa i, que possui um peso

computacional x_i. Um modelo é aplicado para cada tipo de recurso computacional. A metodologia proposta foi testada em uma aplicação baseada em tarefas que realiza a fatoração QR de matrizes esparsas [Buttari 2013], usando a biblioteca StarPU. Esta aplicação divide a fatoração em subproblemas, que são formados por submatrizes menores e mais densas. Cada submatriz é particionada em blocos, sobre os quais se executa um conjunto de tarefas. Um mesmo tipo de tarefa, então, pode ter um custo computacional diferente, já que a esparsidade dos blocos onde é realizada a computação varia, determinando o número de operações de ponto flutuante necessárias.

Resultados e Conclusão: Para representação destas tarefas na ferramenta StarVZ, um painel de visualização que gera um gráfico de Gantt foi incrementado com a informação das tarefas anômalas. Estas tarefas são representadas por cores mais fortes, de forma a destacá-las. Com isto, é possível perceber se a ocorrência de anomalias está associada a algum recurso computacional ou momento específico. Isso é ilustrado pela Figura 1, onde a altura no eixo Y representa o núcleo de CPU em que a tarefa foi executada, e as cores o tipo de tarefa para um intervalo de tempo da aplicação. Podemos ver em alguns casos, que a linha que representa as dependências de uma tarefa anômala, aponta para tarefas executadas em outros núcleos de CPU, o que pode aumentar o número de cache *misses*. Esta informação sobre as tarefas, pode ajudar tanto desenvolvedores de aplicações, quanto quem desenvolve bibliotecas que suportam tal paradigma a investigar e detectar problemas de desempenho. O material usado neste trabalho está publicamente disponível em `https://gitlab.com/mmiletto/erad_2020`.

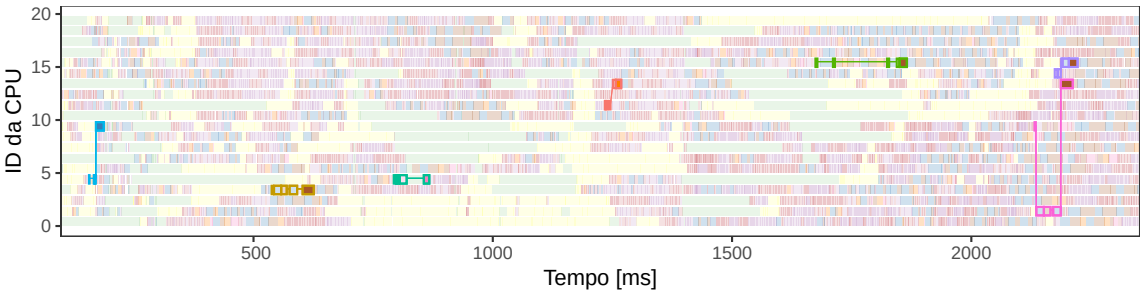

Figura 1. Tarefas anômalas e suas dependências destacadas em um gráfico da ferramenta StarVZ.

Referências

Augonnet, C. et al. (2011). Starpu: a unified platform for task scheduling on heterogeneous multicore architectures. *CCPE*, 23(2):187–198.

Buttari, A. (2013). Fine-grained multithreading for the multifrontal qr factorization of sparse matrices. *SIAM Journal on Scientific Computing*, 35(4):C323–C345.

Dongarra, J. et al. (2017). With extreme computing, the rules have changed. *Computing in Science & Engineering*, 19(3):52.

Garcia Pinto, V. et al. (2018). A visual performance analysis framework for task-based parallel applications running on hybrid clusters. *CCPE*, 30(18):e4472.

Miletto, M. C. and Schnorr, L. (2019). Openmp and starpu abreast: the impact of runtime in task-based block qr factorization performance. In *Anais do XX Simpósio em Sist. Comp. de Alto Desempenho*, pages 25–36. SBC.