Análise de Correlação no Esforço de Desenvolvimento de Aplicações Paralelas

Eduardo Eichner, Gabriella Andrade, Dalvan Griebler, Luiz Gustavo Fernandes[1]

[1] Escola Politécnica, Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Porto Alegre – RS – Brasil

{Eduardo.Eichner, gabriella.andrade}@edu.pucrs.br,

{dalvan.griebler,luiz.fernandes}@pucrs.br

Resumo. *Neste trabalho, foram analisadas diversas métricas, voltadas para uma aplicação de processamento de vídeo, utilizando as interfaces FastFlow, TBB e SPar. Os resultados revelam que utilizando a SPar e o FastFlow é possível desenvolver uma aplicação paralela eficiente com menos esforço, ao contrário do TBB. Em trabalhos futuros planejamos incluir mais aplicações no dataset a fim de confirmar os resultados.*

1. Introdução

O desenvolvimento de aplicações paralelas é uma tarefa complexa para os programadores quando comparada com a programação sequencial. Neste tipo de desenvolvimento, o programador deve se preocupar com diferentes aspectos de paralelismo, incluindo a implementação da sincronização dos dados, divisão do problema de computação entre as *threads*, exploração da concorrência e fornecimento de otimização de *hardware* de baixo nível [McCool et al. 2012]. A fim de facilitar o desenvolvimento de aplicações paralelas, diferentes Interfaces de Programação Paralela (IPPs) foram criadas ao longo dos anos. A maioria dos pesquisadores da área se concentra na avaliação do desempenho das IPPs propostas ao paralelizar uma aplicação sem considerar o esforço requerido no desenvolvimento, tornando difícil determinar qual IPP permite melhor produtividade. Outros trabalhos na área utilizam métricas de codificação para avaliar o esforço de desenvolvimento, como COCOMO II, Halstead, e complexidade ciclomática [Fernàndez et al. 2019, Griebler et al. 2014, Andrade et al. 2021]. Entretanto, essas métricas foram estabelecidas para avaliação de software sem considerar o paralelismo, embora sejam bem estabelecidas na área de Engenharia de Software.

Devido às limitações encontradas nas métricas de codificação existentes, este trabalho tem como objetivo avaliar um conjunto de métricas que acreditamos impactar no esforço necessário para desenvolver aplicações paralelas. Uma vez que realizar otimizações no código a fim de obter desempenho exige esforço da parte do programador, essas métricas incluem o tempo de execução, número de *threads*, e *Speedup* [Andrade et al. 2022]. Sendo assim, neste trabalho avaliamos a correlação das métricas medidas com o tempo de desenvolvimento a fim de avaliar o impacto das mesmas no desenvolvimento de uma aplicação de processamento de *stream* utilizando as interfaces FastFlow, TBB e SPar. Para isso, consideramos os dados de experimento com desenvolvedores iniciantes no domínio da programação paralela [Andrade et al. 2023].

Este trabalho está organizado da seguinte maneira. A Seção 2 contém a metodologia utilizada para a coleta das métricas a serem avaliadas. A Seção 3 apresenta os resultados obtidos e, as conclusões estão descritas na Seção 4.

2. Metodologia

Para a realização deste trabalho, foi desenvolvido um *dataset* composto de diversas informações sobre uma aplicação de processamento de vídeo OpenCV escrita em C++, cujo objetivo é extrair um dos canais RGB de um vídeo. A coleta dos dados foi realizada através de um experimento realizado com 15 estudantes de pós-graduação iniciantes em programação paralela [Andrade et al. 2023]. A tarefa de cada um deles foi paralelizar a aplicação de extração do canal RGB utilizando três IPPs diferentes: FastFlow, TBB e SPar. As três IPPs foram desenvolvidas para a linguagem C++, utilizam o paradigma de programação paralela estruturada e são voltadas para o paralelismo de *stream*. Os 15 participantes foram divididos em três grupos e variou-se a ordem das IPPs utilizadas: Grupo 1 usou SPar, TBB e FastFlow; o grupo 2 usou TBB, SPar e FastFlow; e o grupo 3 usou SPar, FastFlow e TBB.

Para compor o *dataset*, inicialmente consideramos um conjunto de métricas que julgamos afetar o tempo de desenvolvimento. As métricas utilizadas foram: **TempoDesen**: tempo, em horas, para desenvolver a aplicação paralela; **TempoExec**: tempo, em segundos, para executar a aplicação paralela; **NThreads**: número de *threads* utilizadas a fim de obter o melhor desempenho; **Speedup**: medida calculada pela divisão do tempo de execução sequencial pelo tempo de execução paralelo; **Eficiência**: medida calculada dividindo o tempo de execução paralelo pelo número de *threads*; **SLOC**: número total de linhas de código; **ASLOC**: número de linhas de código adicionadas após a paralelização.

Para realizar a análise de correlação foi utilizada a Correlação de pearson (r), uma medida que associa o grau de relacionamento entre duas variáveis. O coeficiente de correlação de pearson varia entre -1 e 1, sendo que o sinal indica, positiva ou negativamente, a direção do relacionamento e o valor mostra a força da relação entre as variáveis. Uma correlação perfeita, -1 ou 1, indica que a relação entre as variáveis são lineares. O valor 0 indica que as variáveis são linearmente independentes e, o valor -0 indica que o coeficiente está negativamente próximo de zero. Os valores de correlação variam conforme a sua classificação ente fraca (0,10 até 0,30), moderada (0,40 até 0,60) ou forte (0,70 até 1). Em resumo, quanto mais perto de 1, independente do sinal, maior é a semelhança entre as variáveis [Figueiredo Filho and Silva Júnior 2009]. Para uma melhor análise, os valores dos gráficos de correlação foram arredondados para uma casa decimal.

3. Resultados

A Figura 1 apresenta o tempo de desenvolvimento, em horas, para cada um dos 15 participantes com o uso de três interfaces (SPar, FastFlow e TBB). Analisando os resultados deste gráfico, é notável que, na maioria dos casos a SPar tem um tempo de desenvolvimento menor se comparado com as outras IPPs (cerca de 53%). Isso ocorre pelo fato da SPar ser baseada em anotações, fazendo com que o desenvolvedor precise modificar o mínimo possível do código para fazer a paralelização. Além disso, alguns participantes tiveram mais facilidade com o FastFlow, pois esta foi a última IPP utilizada por eles. Logo, os participantes já tinham conhecimento de quais regiões do código deveriam paralelizar quando utilizaram o FastFlow, reduzindo o esforço de desenvolvimento.

As Figuras 2a, 2b e 2c apresentam os gráficos de correlação para os resultados da SPar, FastFlow e TBB. Analisando os gráficos de correlação, em relação ao tempo de desenvolvimento, é perceptível que as três interfaces tiveram coeficientes de correlação

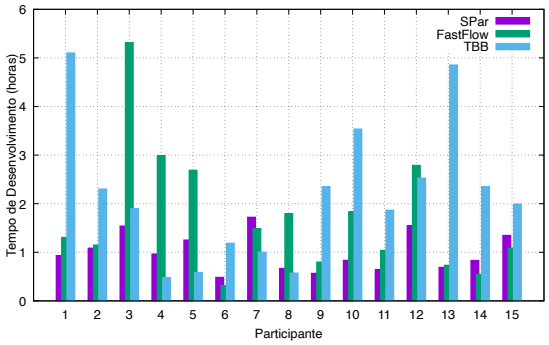
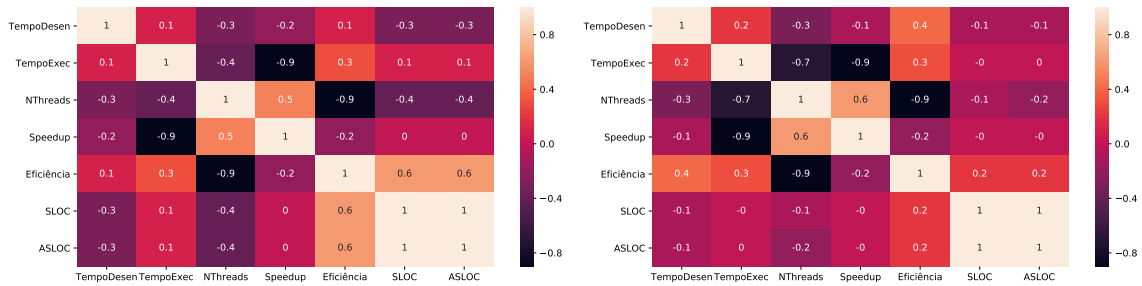


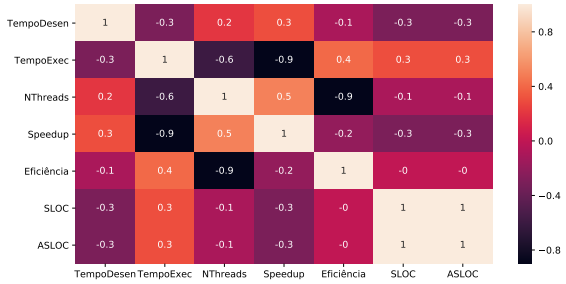
Figura 1. Tempo de desenvolvimento para cada um dos participantes.

diferentes. Para a SPar, o tempo de desenvolvimento apresentou uma correlação fraca (0,1) com as métricas tempo de execução e eficiência. As métricas de número de *threads*, *Speedup*, SLOC e ASLOC tiveram uma correlação negativa em relação ao tempo de desenvolvimento. Isso significa que, quando o valor dessas variáveis aumenta, o esforço de programação tende a diminuir. Logo, os participantes que desenvolveram as versões com melhor desempenho as fizeram com menos esforço de programação. Esses resultados ocorreram devido devido à maneira como o experimento foi conduzido, onde esses participantes iniciaram suas atividades com o TBB e depois a SPar. Além disso, a SPar, em comparação com as outras IPPs, obteve correlação moderada (0,6) entre a Eficiência e as medidas SLOC e ASLOC. Isso se deve pelo fato da SPar precisar adicionar poucas linhas de código para a paralelização.

Para o FastFlow, o tempo de desenvolvimento apresentou uma correlação mode-



(a) Correlação SPar **(b)** Correlação FastFlow



(c) Correlação TBB

Figura 2. Gráficos de Correlação.

rada (0,4) com a Eficiência e fraca (0,2) com o tempo de execução. A correlação entre o tempo de desenvolvimento e o restante das métricas foi negativa. Além disso, o FastFlow apresentou correlação moderada (0,6) entre o número de *threads* e *Speedup*. Isso ocorre por conta da forma que foi realizada o experimento, onde a maioria dos participantes usou o FastFlow depois das outras IPPs. Isso resulta em uma correlação moderada (0,5) entre o número de *threads* e *Speedup* para a SPar e TBB. Além disso, o tempo de desenvolvimento do TBB teve correlação fraca em relação ao *Speedup* (0,3) e ao número de *threads* (0,2). Para as demais métricas essa correlação é negativa. Isso significa que foi necessário mais esforço para desenvolver aplicações com um bom desempenho usando o TBB.

4. Conclusão

Neste trabalho, foram descritas as métricas analisadas juntamente com a aplicação de extração do canal RGB e IPPs utilizadas. Através dos resultados obtidos, pode-se notar que a maioria das métricas avaliadas possui correlação negativa com o tempo de desenvolvimento. Isso ocorreu devido à maneira como o experimento foi conduzido, onde os participantes iniciaram a paralelização da aplicação com TBB e SPar, e finalizaram com FastFlow. Ao utilizar SPar e FastFlow os participantes já haviam adquirido conhecimento a respeito da aplicação a ser paralelizada. Logo, com menos esforço eles conseguiram desenvolver aplicações paralelas com mais desempenho em comparação com o TBB. Em trabalhos futuros pretendemos realizar novos experimentos para contornar as limitações encontradas. Além disso, pretendemos incluir mais aplicações de diferentes tipos de paralelismo (paralelismo de dados e tarefas) e outras arquiteturas (GPU e *cluster*). Por fim, nós pretendemos incluir outras linguagens de programação, como Java utilizando Apache Spark e Apache Storm, e IPPs não estruturadas, como OpenMP e Pthreads.

Referências

Andrade, G., Griebler, D., Santos, R., Danelutto, M., and Fernandes, L. G. (2021). Assessing Coding Metrics for Parallel Programming of Stream Processing Programs on Multi-cores. In *Euromicro SEAA 2021*, pages 291–295, Pavia, Italy. IEEE.

Andrade, G., Griebler, D., Santos, R., and Fernandes, L. G. (2022). Opinião de Brasileiros Sobre a Produtividade no Desenvolvimento de Aplicações Paralelas. In *WSCAD 2022*, Florianópolis, Brasil. SBC.

Andrade, G., Griebler, D., Santos, R., and Fernandes, L. G. (2023). A Parallel Programming Assessment for Stream Processing Applications on Multi-core Systems. *Computer Standards & Interfaces*, 84:103691.

Fernàndez, J. F., Escribano, A. G., and Llanos, D. R. (2019). Simplifying the Multi-GPU Programming of a Hyperspectral Image Registration Algorithm. In *2019 International Conference on High Performance Computing & Simulation*, pages 11–18. IEEE.

Figueiredo Filho, D. B. and Silva Júnior, J. A. (2009). Desvendando os Mistérios do Coeficiente de Correlação de Pearson (r). *Revista Política Hoje*, 18(1):115–146.

Griebler, D., Adornes, D., and Fernandes, L. G. (2014). Performance and Usability Evaluation of a Pattern-Oriented Parallel Programming Interface for Multi-Core Architectures. In *SEKE 2014*, pages 25–30. Knowledge Systems Institute Graduate School.

McCool, M., Reinders, J., and Robison, A. (2012). *Structured Parallel Programming: Patterns for Efficient Computation*. Elsevier.