# Grau de Paralelismo Adaptativo na DSL SPar

**Adriano Vogel, Luiz Gustavo Fernandes**

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS),
Grupo de Modelagem de Aplicações Paralelas (GMAP), Porto Alegre – RS – Brasil

adriano.vogel@acad.pucrs.br

***Resumo.*** *As aplicações de stream apresentam características que as diferem de outras classes de aplicações, como variação nas entradas/saídas e execuções por períodos indefinidos de tempo. Uma das formas de responder a natureza dinâmica dessas aplicações é adaptando continuamente o grau de paralelismo. Nesse estudo é apresentado o suporte ao grau de paralelismo adaptativo na DSL (Domain-Specific Language) SPar.*

## 1. Introdução

A demanda por processar cargas dinâmicas em tempo real trouxe o paradigma de *streams*, sendo aplicações com características únicas [Andrade et al. 2014], como processamento contínuo e com variações constantes no tráfego. Diversas áreas precisam coletar e analisar dados produzidos por diversos dispositivos (ex: câmeras, radares). Objetivos frequentes de desempenho nessas aplicações são alto *throughput* e baixas latências.

Executar em paralelo é uma forma de oferecer ganhos de desempenho para aplicações de *stream*. Porém, a complexidade de programação somada a necessidade de conhecimento da arquitetura dificultam a tarefa de exploração do paralelismo. SPar [Griebler et al. 2017] é uma DSL interna para paralelismo de *stream* que busca aumentar a produtividade de código usando anotações e atributos no código sequencial, fazendo com que os programadores apenas anotem potenciais regiões a serem paralelizadas. Com essas anotações, o compilador da SPar reconhece os atributos anotados e gera código paralelo com o auxílio da biblioteca FastFlow. O paralelismo é explorado na forma de pipeline que pode conter estágios replicados com um grau de paralelismo fixo que é definido pelo programador [Griebler et al. 2017].

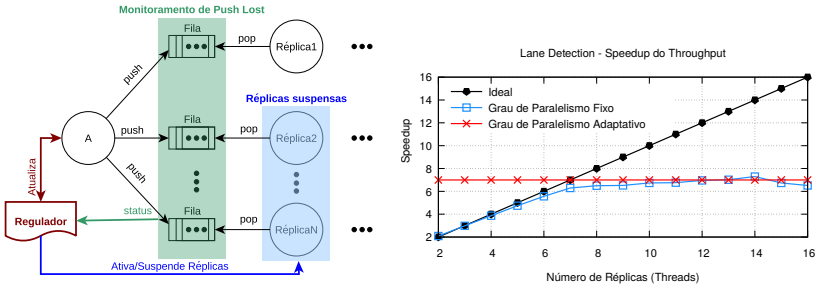
O objetivo deste trabalho é evitar que o desenvolvedor necessite especificar o grau de paralelismo adequado para uma aplicação de processamento de *stream* na DSL SPar. Além de abstrair, a dinamicidade também se faz necessária neste tipo de aplicação, pois a execução é normalmente indefinida e as cargas de trabalho variam durante a execução. Desta forma, o grau de paralelismo deve ser adaptativo para manter um bom desempenho e utilizar os recursos computacionais de forma eficiente.

Este trabalho compartilha características com outros estudos, como usar o *framework* FastFlow [Sensi et al. 2016], e o cenário de *Stream* com [Gedik et al. 2014]. Contrastando com os estudos relacionados que trazem algoritmos com execução adaptativa, esse trabalho abstrai dos programadores a necessidade de definir o grau de paralelismo. O objetivo futuro é gerar código paralelo na DSL SPar com essa funcionalidade.

## 2. Paralelismo Adaptativo

Grau de paralelismo adaptativo ocorre através de um mecanismo que monitora a *runtime* durante a execução. Tal mecanismo decide sobre o aumento ou a diminuição do grau

de paralelismo. Na SPar, o grau de paralelismo é o número de réplicas de um estágio em uma região de paralelismo de *stream*. O mecanismo implementado adapta o grau de paralelismo considerando estatísticas de congestionamento (usando um *threshold*) nas filas das *threads* de processamento, como mostrado na Figura 1(a).




(a) Mecanismo para Adaptação do Paralelismo.

(b) Speedup do Throughput.

**Figura 1. Resultados.**

O mecanismo foi avaliado[1] em uma aplicação de vídeo que faz a detecção de pistas em rodovias e usando um vídeo de *input* de 5.25MB (640x360 pixeis) . O resultado do *throughput* de execução do programa paralelo com grau de paralelismo adaptativo foi comparado com execuções do paralelismo no modo fixo e com o sequencial. A Figura 1(b) apresenta o *speedup* do *throughput* das execuções paralelas. O modo adaptativo usa um número variante de réplicas de acordo com os congestionamentos das filas das *threads* de processamento enquanto o fixo usa o mesmo grau de paralelismo na execução. O melhor *speedup* da aplicação foi com 14 réplicas no modo fixo, pois a escalabilidade da aplicação é limitada. O resultado usando um grau de paralelismo adaptativo foi próximo ao melhor caso do fixo e melhor que as demais execuções. Além disso, tornar o grau de paralelismo adaptativo considerando um objetivo de desempenho de *throughput* e latência está sendo implantado na SPar bem como a avaliação do *overhead* dos mecanismos que adaptam o grau de paralelismo.

## Referências


[Andrade et al. 2014] Andrade, H., Gedik, B., and Turaga, D. (2014). *Fundamentals of Stream Processing: Application Design, Systems, and Analytics*. Cambridge University Press.

[Gedik et al. 2014] Gedik, B., Schneider, S., Hirzel, M., and Wu, K.-L. (2014). Elastic scaling for data stream processing. *IEEE Transactions on Parallel and Distributed Systems*, 25(6):1447–1463.

[Griebler et al. 2017] Griebler, D., Danelutto, M., Torquati, M., and Fernandes, L. G. (2017). SPar: A DSL for High-Level and Productive Stream Parallelism. *Parallel Processing Letters*, 27(01):20.

[Sensi et al. 2016] Sensi, D. D., Torquati, M., and Danelutto, M. (2016). A reconfiguration algorithm for power-aware parallel applications. *ACM Transactions on Architecture and Code Optimization (TACO)*, 13(4):43.


[1]Nodo com 8 cores (16 threads) e 24GB de RAM