Provendo Abstrações de Alto Nível para GPUs na SPar

Dinei André Rockenbach[1], **Dalvan Griebler**[1], **Luiz Gustavo Fernandes**[1]

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

dinei.rockenbach@edu.pucrs.br,dalvan.griebler@pucrs.br

Resumo. *O presente trabalho apresenta uma extensão à linguagem SPar para suportar o paralelismo heterogêneo combinado de CPU e GPU através de anotações C++11 em aplicações de processamento de stream. Os testes sugerem melhoras significativas de desempenho com poucas modificações no código.*

1. Estudo e Pesquisa

Os novos hardwares surgidos nos últimos anos têm seguido a tendência crescente no volume de dados gerados pelo uso das tecnologias digitais. Na vanguarda da busca por mais desempenho estão as *Graphics Processing Units* (GPUs), hardwares massivamente paralelos originalmente desenhados para processamento gráfico, mas que hoje em dia são largamente utilizados em várias tarefas de cunho geral. O advento das GPUs impulsionou aplicações como carros com direção autônoma, *ray tracing* em tempo real, aprendizado profundo na inteligência artificial e realidade virtual (VR). Porém, esse ambiente heterogêneo com GPUs e *Central Processing Units* (CPUs) paralelas apresenta um desafio adicional para o desenvolvimento de software paralelo.

A programação estruturada é uma alternativa para facilitar o desenvolvimento dessas aplicações através do uso de padrões paralelos. Estes padrões simplificam a programação utilizando estruturas algorítmicas (*algorithmic skeletons*) com fluxos de execução paralelos pré-programados. Porém, para a exploração de paralelismo em GPU atualmente tanto a academia quanto a indústria usam CUDA e OpenCL, que não oferecem interfaces de programação estruturada e cujos desenvolvedores precisam lidar com conceitos de arquitetura de hardware de baixo nível. Existe uma carência de abstrações de programação paralela de alto nível que lidam com o paralelismo de CPU e GPU combinados.

Muitas aplicações que desejam realizar processamento na GPU são aplicações de processamento de *stream*, que computam uma sequência de operadores sobre dados que são gerados em fluxo contínuo. Cada um dos operadores pode processar diferentes dados simultaneamente, que é o paralelismo de *stream*. Esse paralelismo é adequado para a CPU, enquanto que o paralelismo de dados é adequado para processamento na GPU. Por exemplo, uma aplicação de geração de vídeo expõe paralelismo de *stream* na sequência de imagens a ser processada, porém cada imagem expõe paralelismo de dados uma vez que a cor de cada pixel pode ser computada individualmente. Este trabalho continua a proposta apresentada em [Rockenbach et al. 2019] para fornecer abstrações de alto nível para a exploração do paralelismo em ambientes heterogêneos com CPUs e GPUs.

A SPar [Griebler et al. 2017] é uma linguagem de domínio específico (*Domain-Specific Language* ou DSL) focada no paralelismo de *stream*. Ela oferece cinco atributos C++11 que podem ser inseridos no código através de anotações. O código anotado é processado por um compilador que efetua transformações *source-to-source*, inserindo chamadas a bibliotecas de nível mais baixo, tais como FastFlow, TBB e OpenMP, para

explorar o paralelismo de *stream* através da geração eficiente dos padrões *pipeline* e *farm*, ou a composição de *pipeline* com *farm*(s). Para permitir a exploração do paralelismo massivo das GPUs utilizando as abstrações de alto nível da SPar, foi necessário estender a linguagem incluindo três novos atributos: (1) `Pure`, para indicar estágios ou blocos de código puros que podem ser enviados para a GPU; (2) `Batch`, para diminuir a granularidade dos elementos do *stream* e fornecer maior volume de dados para cada chamada à GPU; e (3) `Reduce`, para indicar operações de reduções dentro de blocos de código puro.

Para suportar estes novos atributos, foram elaboradas regras de transformação *source-to-source*, que foram implementadas no compilador a fim de gerar os padrões paralelos *map* e *map-reduce* utilizando a biblioteca GSParLib [Rockenbach 2020]. Essa biblioteca foi desenvolvida com uma interface de programação estruturada unificada e um ambiente de execução agnóstico ao *driver* da plataforma de hardware. Ela oferece os padrões paralelos *map* e *reduce* sobre os *drivers* CUDA e OpenCL.

Os testes de desempenho da aplicação Lane Detection foram executados sobre 3.169 *frames* com resolução 1242x375 do conjunto de dados KITTI e foram executados em uma máquina com CPU Intel® Core™ I9-7900X @ 3.3 GHz e uma GPU Titan Xp com 12 GB GDDR5X @ 2400 MHz de memória. Partindo do código sequencial, foram necessárias apenas três anotações da SPar para explorar o paralelismo de CPU e outras três anotações para explorar o paralelismo heterogêneo combinado de CPU e GPU. Na Figura 1 é possível observar que a SPar apresentou melhoras de desempenho significativas, com 7.9× *speedup* e 255 FPS na versão multi-core (em verde) com nove *workers*, 11.5× *speedup* e 374 FPS na versão GPU com OpenCL (em laranja) e 10.7× *speedup* e 348 FPS na versão GPU com CUDA (em azul). Pode-se notar que quando há mais de 5 *workers* nas versões que utilizam a GPU, o gargalo da aplicação passa a ser esse processador, uma vez que os testes foram realizados com uma única GPU.

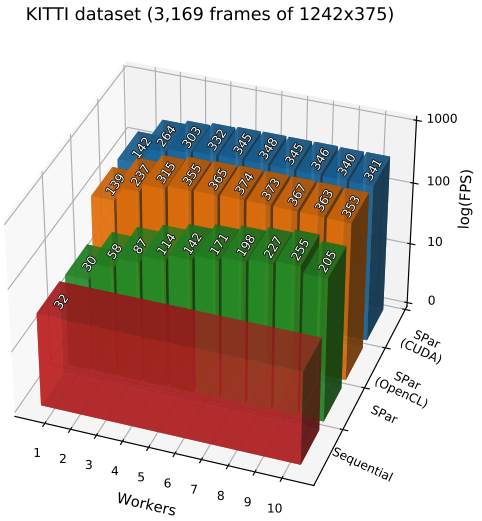

Figura 1. *Throughput* do Lane Detection.

Com essa extensão, a linguagem SPar oferece suporte ao paralelismo de dados em GPU e paralelismo de *stream* em CPU, permitindo explorar eficientemente o paralelismo heterogêneo em aplicações de processamento de *stream* com abstrações de alto nível.

Referências

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):1740005.

Rockenbach, D. A. (2020). High-Level Programming Abstractions for Stream Parallelism on GPUs. Master's thesis, School of Technology - PPGCC - PUCRS, Porto Alegre, Brazil.

Rockenbach, D. A., Griebler, D., and Fernandes, L. G. (2019). Proposta de Suporte ao Paralelismo de GPU na SPar. In *Escola Regional de Alto Desempenho (ERAD-RS)*, page 4, Três de Maio, BR. Sociedade Brasileira de Computação (SBC).