Avaliação de Arquiteturas GPU para Implementação de Técnicas de Tolerância a Falhas em Sistemas Embarcados

Caroline Zingano de Aguiar*, Luigi Carro
{czaguiar, carro}@inf.ufrgs.br
***Bolsista IC – Programa FP7 da Comissão Européia**

Objetivo

Avaliar o desempenho de arquiteturas GPU (Unidade de Processamento Gráfico) para implementação de técnicas de tolerância a falhas de hardware via software em sistemas embarcados.

Motivação

GPUs possuem hardware ocioso, pois aplicações não usam todo o recurso de paralelismo existente. É possível utilizar esses recursos a fim de oferecer tolerância a falhas com baixo impacto em performance.

Problema e Solução

- Neste trabalho, a técnica de tolerância a falhas utilizada (Freivalds) envolve operações algébricas, como multiplicações de matrizes e vetores;
- Essas operações processam elevadas quantidades de dados e tornam-se onerosas para os sistemas embarcados atuais;
- Mas possuem bastante paralelismo para ser explorado, permitindo amortizar as operações .

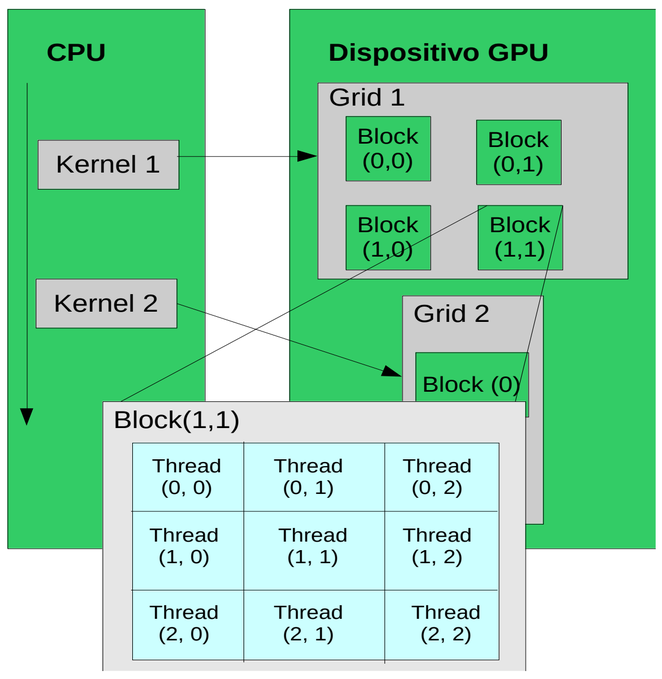


Figura 1: Arquitetura de execução de software em CUDA

- GPUs possuem vários núcleos para processamento em paralelo;
- Operações são particionadas e cada componente é excutado em uma thread;
- Por exemplo: em uma multiplicação de matrizes quadradas, na CPU, NxN operações serão efetuadas, uma por vez; na GPU, as NxN multiplicações são computadas ao mesmo tempo, uma em cada thread.

Estudo de Caso e Metodologia

Algoritmos utilizados:

- Multiplicação de Matrizes;
- IMDCT (Transforma Discreta Inversa do Cosseno): detecção e correção de erros a partir de multiplicações de matrizes e vetores;
- DFT (Transformada Discreta de Fourier): multiplicação de matrizes com elementos complexos.

Dimensões de Vetores e Matrizes utilizadas:

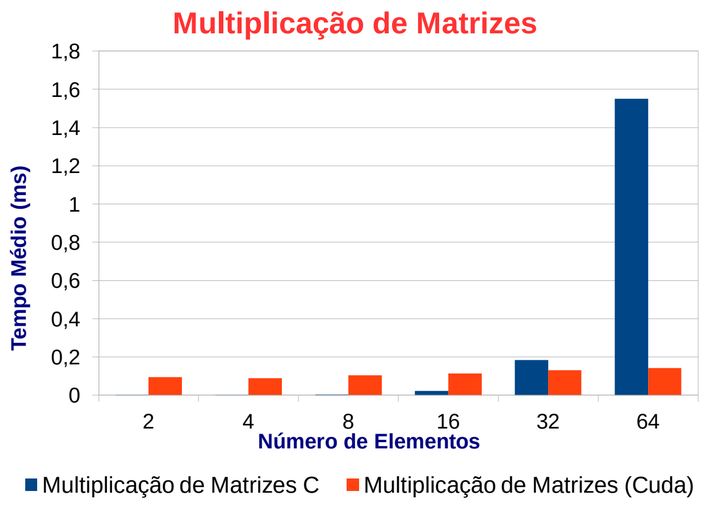
- Multiplicação de Matrizes: matrizes N x N;
- IMDCT : Matrizes N x N/2, N/2 x N, N/2 x N/2 e vetores N;
- DFT: Matrizes N x N e Vetores N;

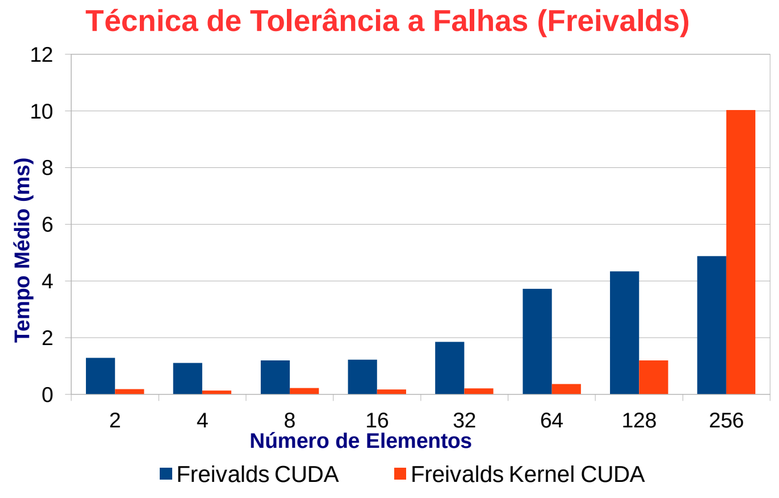
(2^i , onde $0 < i \leq 13$)

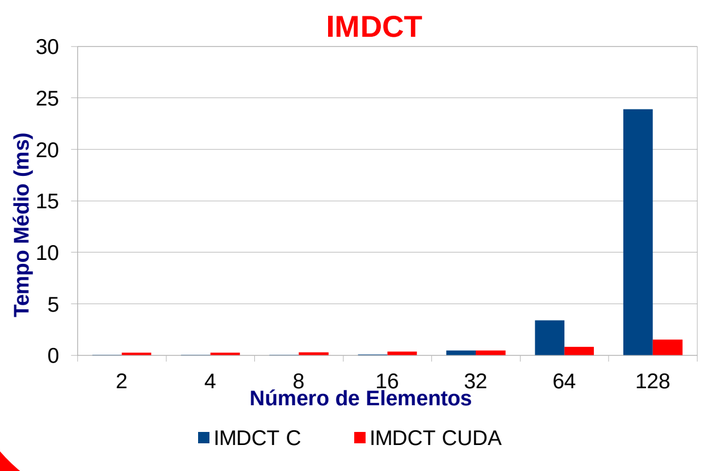
Cenários de execução para cada algoritmo:

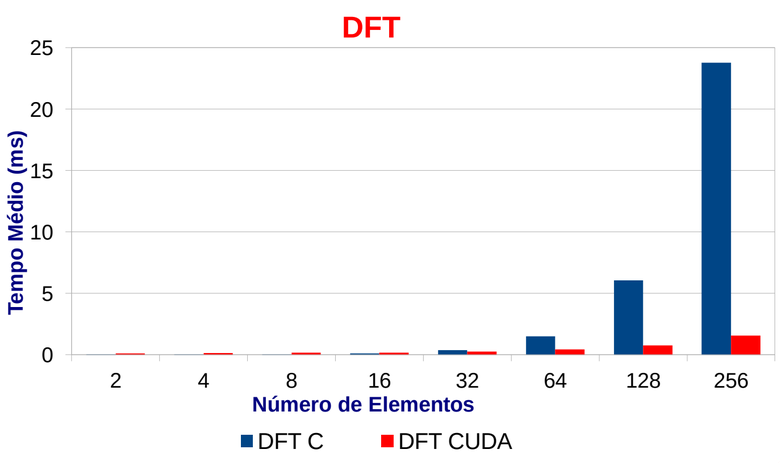
- Execução em GPU: os algoritmos codificados na linguagem de programação CUDA;
- Execução em CPU: algoritmos codificados na linguagem de programação C;
- Técnica de tolerância a falhas: duas versões foram executadas na GPU: uma com várias chamadas de funções e outra com um uníco kernel.

Resultados Experimentais









Resultados e Conclusões

- O uso de GPUs favorece a implementação de técnicas de tolerância a falhas de hardware implementadas em software.
- Alto desempenho amortiza o tempo de execução de algoritmos de proteção contra falhas.
- Nos experimentos de Multiplicação de Matrizes, no caso de matrizes 128 x 128 a execução na CPU foi 7.000 % mais lenta do que na GPU.
- Na execução do algoritmo IMDCT, a versão testada na GPU foi 400.000% mais rápida do que testada na CPU.
- Para a técnica de Freivalds foram testadas duas versões na GPU: a implementada em um único kernel apresentou desempenho superior para valores de N menores do que 256.

Universidade Federal do Rio Grande do Sul
Instituto de Informática – Laboratório de Sistemas Embarcados
http://www.inf.ufrgs.br/~lse