Desempenho dos algoritmos Blowfish e RC6 usando CUDA

Luccas Quadros[1], Roberta Spolon[2], Renata S. Lobato[3], Aleardo Manecero Jr[3]

[1]Bacharelado em Ciência da Computação, UNESP, Bauru - SP,

[2]Departamento de Computação UNESP, Bauru - SP,

[3]Departamento de Computação e Estatística, UNESP São José do Rio Preto – SP

{luccas.quadros,roberta}@fc.unesp.br, {renata, aleardo}@ibilce.unesp.br

Abstract. *Parallel computing is able to guarantee a much faster performance when compared to the sequential versions of the algorithms, which has been much explored in research in several areas. In cryptography, where more and more data needs to be encrypted securely and efficiently, graphics cards appear as a cost-effective alternative for running these algorithms in parallel. This study examines the performance of Blowfish and RC6, two efficient and popular cryptographic algorithms, in their sequential, C ++ and parallel versions, using the GPU through the CUDA programming model and compares the performance achieved.*

Resumo. *A computação paralela pode possibilitar um melhor desempenho se comparada as versões sequenciais dos algoritmos, o que vem sendo muito explorado em pesquisas em diversas áreas. Na criptografia, onde cada vez mais uma quantidade maior de dados precisa ser cifrada de forma segura e eficiente, as placas gráficas surgem como uma alternativa de bom custo-benefício para a execução destes algoritmos em paralelo. Este estudo analisa o desempenho do Blowfish e do RC6, dois algoritmos eficientes e populares de criptografia, em suas versões sequenciais, em C++ e paralelas, utilizando a GPU através do modelo de programação CUDA. Os resultados mostram a vantagem computacional proporcionada por algoritmos paralelos.*

Introdução

Os algoritmos criptográficos são amplamente utilizados para manter um nível de segurança nas interações pela internet. A criptografia simétrica e assimétrica colabora para que as pessoas se sintam mais seguras ao conversar, comprar e navegar pela rede. Em tempos de computação em nuvem, a transparência, para o usuário, das operações realizadas pelos servidores é essencial e para isso é necessário que a velocidade da cifragem e decifragem das informações seja cada vez maior [Lee et al. 2014].

As CPUs já não são capazes de acompanhar esse ritmo de aceleração, o que gera a exploração das GPUs poderosas e com bom custo benefício e consequentemente, do paralelismo destes algoritmos de criptografia. Este estudo, explora o paralelismo de dois algoritmos de criptografia simétrica, Blowfish e RC6, ambos implementados no modo Eletronic Code Book (ECB), que favorece a paralelização das operações.

O artigo está organizado da seguinte forma: A seção 1 descreve brevemente os algoritmos de cifragem de bloco e caracteriza os algoritmos escolhidos para a implementação. A seção 2 detalha o uso de GPUs para algoritmos de uso geral e a ferramenta CUDA. A seguir na seção 3, a metodologia utilizada é exposta e em seguida na seção 4 os resultados. Finalmente apresentamos nossa conclusão na seção 5.

1. Block Ciphers

Em criptografia, uma cifra de bloco é uma operação de cifragem de chave simétrica, sobre blocos de dados de tamanho fixo. Existem diversos algoritmos de cifragem de bloco aplicados em protocolos criptográficos. Neste artigo selecionamos dois algoritmos, amplamente utilizados, como alvo de nossas comparações o Blowfish [Schneier 1993] e o RC6 [Rivest et al. 1998]. Para as implementações utilizamos o modo Eletronic Code Book (ECB), neste modo cada bloco de dados é processado independentemente, o que garante um bom nível de paralelismo.

1.1. Blowfish

Segundo seu criador [Schneier 1993] o Blowfish é uma cifra de bloco de 64-bits e chave qualquer tamanho até 448 bits. Ainda que exija uma complexa fase de inicialização antes de qualquer encriptação, a encriptação em si é rápida e funciona de forma eficiente em alguns microprocessadores. Ainda segundo [Schneier 1993] o algoritmo ocorre com a utilização de uma rede de Feistel de 16 rodadas, na qual cada rodada consiste em uma permutação dependente de chave e uma substituição dependente de chave e dados. Todas as operações são XORs e adições em palavras de 32 bits. As únicas operações adicionais são quatro pesquisas de dados de matriz indexadas por rodada.

1.2. RC6

Segundo seus criadores [Rivest et al. 1998] RC6 é uma melhoria evolutiva do RC5, projetada para atender aos requisitos do Padrão de Cifração Avançado (AES). O algoritmo tem blocos de 128-bits e suporta chaves de 128, 192 e 256-bits até 2040-bits. Assim como o RC5, pode ser parametrizado para suportar uma grande variedade de tamanhos de palavras e rodadas, e portanto [Rivest et al. 1998] explica que o algoritmo é mais precisamente especificado como RC6-w/r/b no qual w representa o tamanho da palavra em bits, r é um número não negativo de rodadas de encriptação e b denota o tamanho da chave em bytes. Para os estudos deste artigo utilizamos a configuração submetida ao AES que tem w=32 e r=20, RC6 apenas é a estenografia desta versão.

2. CUDA e GPU

Impulsionadas pela indústria de jogos, as GPUs se desenvolveram rapidamente nas últimas décadas se tornando uma massiva plataforma de computação paralela, com milhares de núcleos. Nos estudos deste artigo utilizamos a placa gráfica NVIDIA GeForce GTX 465 que conta com 352 CUDA Cores. CUDA é um ambiente de desenvolvimento lançado pela NVIDIA para facilitar a programação geral de GPU que suporta C / C ++. Para utilizar completamente a potência de processamento da GPU, precisamos iniciar um grande pool de threads. Este grupo de threads é ainda dividido em blocos de thread menores; cada bloco de thread contém vários threads.

3. Metodologia

As implementações, tanto para a CPU quanto para a GPU, não diferem em seus algoritmos de cifragem e decriptação. Todavia, na GPU é necessário primeiramente mover os dados da memória principal para a memória da GPU e, após a computação, liberar o resultado da memória da GPU de volta para a memória principal, um custo em termos de desempenho.

Os dois algoritmos compartilham de características semelhantes, uma delas é a necessidade de alguma estrutura de dados que deve ser carregada previamente, estas estruturas são resultado de operações ou expansões com a chave criptográfica que aqui foram executadas exclusivamente na CPU e não são consideradas para o desempenho neste estudo. Após calculadas estas estruturas estão prontas para serem usados na cifragem e decifragem de todos os blocos e, portanto também devem ser enviadas para a GPU para a execução das operações paralelas.

Essas operações, anteriores a criptografia em si, mas necessárias no uso de GPUs, afetam diretamente o desempenho final da execução e portanto, foram considerados neste estudo os tempos de alocação e cópia de memória, além do tempo de execução do algoritmo.

Ambas implementações paralelas seguem o mesmo padrão. Para a cifragem,o texto claro é carregado na memória em palavras, de 32 bits para ambos os algoritmos, juntamente com a tabela de dados necessária. A partir daí são criados o número de threads necessárias para executar esses blocos. No blowfish cada thread trabalha com duas palavras, formando o bloco de 64 bits, já no RC6 os blocos são de 128 bits portanto 4 palavras por thread. Após a cifragem os blocos são copiados para a memória principal. Como o modo escolhido para as implementações foi o ECB estes blocos são apenas concatenados. Para a decriptografia o processo é semelhante, desta vez entrando com o texto cifrado e utilizando o algoritmo correspondente.

4. Resultados

Os algoritmos foram testados em um ambiente com as seguintes configurações:

- Processador Intel® Core™ i7-870 de 2,93 GHz e 8 GB de memória RAM.
- GPU NVIDIA GeForce GTX 465: 1024MB de memória dedicada, 352 CUDA Cores e 102,59 GB/s de largura de banda de memória.

Utilizamos em todos os testes a capacidade máxima de threads por bloco, 1024. Para os testes foram usados arquivos de textos claros gerados aleatoriamente que foram cifrados e decifrados utilizando exatamente a mesma função nos algoritmos sequenciais e paralelos, isto é possível devido ao uso de ambas as tags, provenientes da ferramenta CUDA, `_host_` e `_device_` nas funções que executam essas operações.

Analisando os resultados apresentados na Figura 4 vemos que para arquivos pequenos os algoritmos possuem vantagem, isso se dá devido a custos de iniciação do algoritmo paralelo já citados anteriormente, o carregamento dos dados e a inicialização das threads.

A razão de desempenho entre o algoritmo paralelo e o sequencial, chamado de speedup, obteve um máximo de 24 para o Blowfish e 17 para o RC6, é interessante analisar que conforme os arquivos se tornam maiores há um aumento constante no tempo

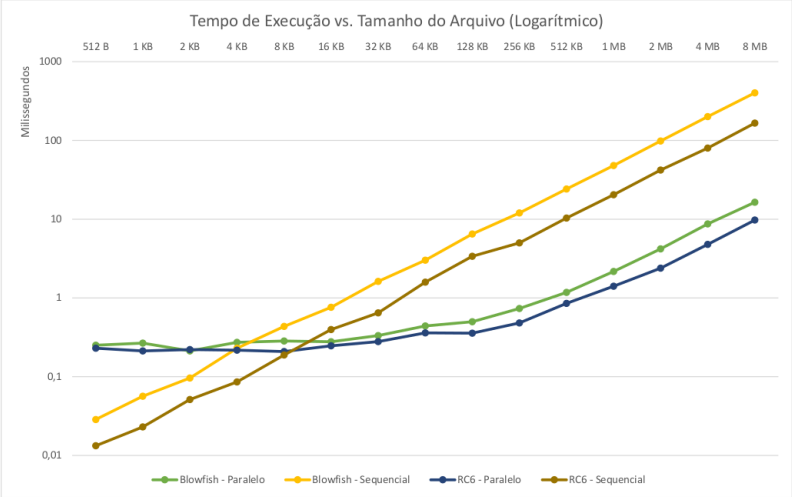

Figura 1. Comparação de desempenho Paralelo vs Sequencial (elaborado pelo autor)

de execução de ambos (CPU e GPU), ainda assim o tempo de execução dos algoritmos paralelos se mantém satisfatoriamente mais rápido.

5. Conclusão

Com os resultados das implementações fica evidente a vantagem computacional proporcionada por algoritmos paralelos, a GPU se mostra, assim sendo, uma boa solução para acelerar os processos de criptografia de cifradores de blocos de chaves simétricas, apresentando resultados até 24 vezes melhores em nossos testes. Ainda que o algoritmo Blowfish tenha apresentado um speedup maior e portanto seja o que mais se beneficia do uso do paralelismo, o RC6 obteve tempos melhores para os mesmos tamanhos de arquivo tanto se comparados os sequências quanto os paralelos.

Referências

Khalifa, O. and Loidl, H.-W. (2011). *The performance of cryptographic algorithms in the age of Parallel computing*. PhD thesis, M. sc thesis, August-2011, Heriot Watt University School Of Mathematical and Computer Science.

Lee, W.-K., Goi, B.-M., Phan, R. C.-W., and Poh, G.-S. (2014). High speed implementation of symmetric block cipher on gpu. In *Intelligent Signal Processing and Communication Systems (ISPACS), 2014 International Symposium on*, pages 102–107. IEEE.

Liu, G., An, H., Han, W., Xu, G., Yao, P., Xu, M., Hao, X., and Wang, Y. (2009). A program behavior study of block cryptography algorithms on gpgpu. In *Frontier of Computer Science and Technology, 2009. FCST'09. Fourth International Conference on*, pages 33–39. IEEE.

Rivest, R. L., Robshaw, M., Sidney, R., and Yin, Y. L. (1998). The rc6tm block cipher. In *First Advanced Encryption Standard (AES) Conference*, page 16.

Schneier, B. (1993). Description of a new variable-length key, 64-bit block cipher (blowfish). In *International Workshop on Fast Software Encryption*, pages 191–204. Springer.