SIMULAÇÃO QUÂNTICA DISTRIBUÍDA VIA GPUS

Anderson Braga de Avila[1]; Renata Reiser[1]; Mauricio Pilla[1]

[1]*Universidade Federal de Pelotas – {abdavila, reiser, pilla}@inf.ufpel.edu.br}*

1. INTRODUÇÃO

A Computação Quântica (CQ), consiste em uma grande mudança de paradigma na ciência da computação. Ele afirma que os algoritmos quânticos podem desenvolver tarefas mais complexas se comparadas com paradigma de programação atual, como no processamento da informação e criptografia (GROVER, 1996; SHOR 1997). Mas enquanto os computadores quânticos ainda não estão disponíveis, o estudo e desenvolvimento de algoritmos quânticos pode ser feito a partir da descrição matemática ou softwares de simulação. Uma vez que a simulação quântica realizado por computadores clássicos exige elevados recursos computacionais (processos e/ou memória) a pesquisa sobre arquiteturas paralelas pode fornecer possíveis melhorias de desempenho para novos algoritmos quânticos.

Neste contexto, este trabalho contribui com o desenvolvimento o Projeto D-GM, disponibilizando um ambiente de modelagem gráfica e um ambiente de execução e gerenciamento de aplicações, denominados VPE-qGM e VirD-GM.

O paradigma de programação GPGPU (*General Purpose Computingon Graphics Processing Units*) tornou-se recentemente uma das abordagens mais interessantes para HPC (*High Performance Computing*) devido ao seu bom equilíbrio entre custo e benefício. O trabalho em HENKEL 2010 explora GPUs para simulação quântica. Neste contexto, tem-se o espaço de memória disponível na GPU como principal restrição, uma vez que o tempo de simulaçãomostra-se bem reduzido. A inovação na área de pesquisa de computação/simulação quântica é integrar num mesmo ambiente essas duas abordagens: simulação distribuída e GPU. Nosso projeto visa consolidar o ambiente D-GM como suporte para esse cálculo e processamento híbrido da simulação quântica.

Este trabalho descreve os resultados obtidos com a integração da biblioteca de execução baseada em GPU para o gerenciador de simulação distribuída VirD-GM. Assim, tem-se os fundamentos da simulação quântica distribuída em GPU, a qual será posteriormente consolidada por ambas estratégias: (i) integração da biblioteca de execução com base OpenMP; (ii) reestruturação do nível de programação e processamento para distribuição de tarefas entre CPUs e GPUs.

O gerenciador de simulação Vird-GM, implementado na linguagem Java, suporta o framework JCUDA, o qual fornece os métodos para transferência de dados entre CPU e GPU. Essa estratégia nos permite realizar a comparação entre o framework JCUDA e PyCUDA, que foi explorado com bons resultados MARON et al. (2012).

2. METODOLOGIA

Este trabalho considera a simulação distribuída de portas Hadamard até 21 qubits. Esta escolha se justifica pois a porta Hadamard representa as operações com o maior custo computacional no VPE-qGM. Os resultados obtidos em AVILA et al. (2012) são utilizadas como referência para comparar o funcionamento da CPU/GPU até 17 qubits. Além disso, uma outra comparação entre implementações considerando os *frameworks* PyCUDA e JCUDA foi realizado.

A simulação distribuída via CPU foi realizada utilizando dois desktops, cada um com quatro *vird-clients*, com a seguinte configuração: processador Intel Xeon E5620, 4 GB de RAM e rede Fast Ethernet. Uma máquina adicional foi necessário para executar o servidor Vird-GM.

A metodologia adotada para a simulação distribuída via CPU considera a execução de 15 simulações de cada instância das transformações Hadamard, considerando cada uma das configurações de processadores dentro do cluster. Durante a simulação via GPU, o tempo médio de simulação foi coletado e os testes realizados em um desktop com processador Intel Core i7-3770, 8 GB de RAM e uma GPU NVIDIA GT640. Como componentes de software, tem-se: PyCuda 2012.1, JCUDA 0.5.0a, NVIDIACUDA 5, NVIDIA VisualProfiler 5.0 e Ubuntu 12.04 64bits. O tempo de simulação foi obtidoapós 10 execuções decada aplicação.

3. RESULTADOS E DISCUSSÃO

Na Tabela 1, tem-se o tempo de simulação para as transformações Hadamard até 21 qubits. O desvio padrão máximo de 1,6% foi medido para a $H^{\otimes 16}$. Como esperado, a simulação via GPU supera a via CPU em todos os cenários em pelo menos 19 vezes. Salienta-se que, quanto maior o número de qubits da simulação, maior é o ganho obtido na simulação baseada em GPU.

Tabela 1: Média do tempo de execução, medido em segundos, para os estudos de caso deste trabalho.

Op	1 VirD-Client		2 VirD-Clients	
	CPU	GPU	CPU	GPU
H^{14}	36.061	1.679	19.293	1.623
H^{15}	138.388	2.083	72.824	1.949
H^{16}	1126.529	2.319	573.611	2.785
H^{17}	3075.729	7.469	1632.369	5.207
H^{18}	NE	23.604	NE	13.995
H^{19}	NE	86.404	NE	47.046
H^{20}	NE	345.433	NE	179.383
H^{21}	NE	1364.844	NE	696.538

NE: Não executado. Requer muito tempo.

Verificou-se ainda que a simulação distribuída via GPU pode tirar vantagem de nós de processamento adicionais, quando a carga de

computação é significativa. Veja na Figura 1, onde uma boamelhora é obtida por sistemas com pelo menos 17 qubits .

Entretanto, salienta-se que nesta fase inicial de desenvolvimento do *kernel* de execução para a GPU ainda não há suporte para portas controladas

Estes resultados nos permite realizar uma comparação entre a simulação quântica distribuída via GPU, proposto neste trabalho, e a única simulação sequencial em GPU usando o framework PyCUDA, como descrito em MARON et al. (2012).

Tendo em conta que a arquitetura do ambiente Vird-GM é mais complexa e os dados devem ser enviados para os Vird-Clients através da rede, existem alguns casos em que a simulação auxiliado pela estrutura PyCUDA supera a simulação quântica distribuída via GPU.

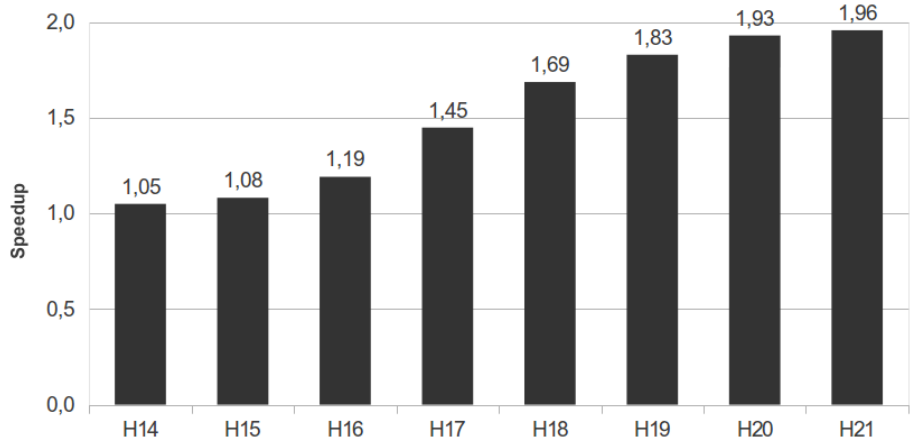

Figura 1: Taxa de escala para 2 vird-clients

A Figura 2 mostra que a simulação distribuída é melhor explorada quando mais de 17 qubits são utilizados. Como as principais operações do kernel CUDA são as mesmas para ambas abordagens, a maior parte da sobrecarga na simulação distribuídase deve à rede de comunicação servidor/cliente.

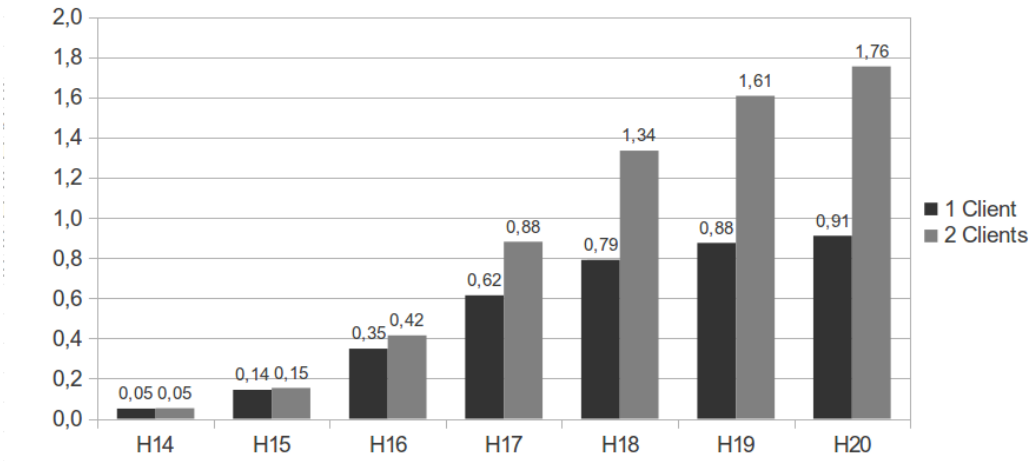

Figura 2: Comparação de desempenho entre simulações no JCUDA (distribuído) e no PyCUDA (nó único)

4. CONCLUSÕES

O *framework* VirD-GM tem como principal objetivo fornecer uma estrutura de suporte ao gerenciamento da simulação quântica distribuída no ambiente D-GM, capaz de explorar os benefícios da computação GPGPU. Caracterizada pela, a nossa contribuição ao Vird-GM viabilizou a conecção dos clientes com a execução em GPUs. Como o kernel CUDA está escrito na linguagem C, e o cliente de execução é escrito em Java, usamos o framework JCUDA.

Os resultados mostram que a simulação quântica distribuída baseada em GPU supera a abordagem baseada em CPU para os casos de Hadamard simulados, com grande significadoparao escopo globalda simulação.Uma vez que o kernel CUDA necessitou de apenas algumas mudanças da sua versão original, não há necessidade de manter várias versões para a execução local em GPU (com PyCUDA) e execução distribuída em GPU (com JCUDA).

Também verificou-se que a simulação distribuída é mais adequada para sistemas com mais de 17 qubits. Devido às operações adicionais executadas pelos módulos de gerenciamento distribuído, o volume de operações a ser executado por cada cliente tem de ser suficientemente grande (pelo menos 17 qubits) para superar essa sobrecarga.

Trabalhos futuros em nosso projeto estão descritos nos seguintes tópicos: (i) redefinição do algoritmo de execução otimizando a distribuição da memória; (ii) suporte para portas controladas, projeções e operações de medição na abordagem distribuída; (iii) concepção e implementaçãodo cliente de execução, em que o cálculo será executado pelo CPU e GPU; (iv) simulações de aplicações distribuídas usando GPUs .

5. REFERÊNCIAS BIBLIOGRÁFICAS

AVILA, A.;MARON, A.;REISER, R.; PILLA, M.Extending the VirD-GM environment for thedistributed execution of quantum processes. In:**Proceedings of XIII WSCAD-WIC**, pages 1–4,2012.

GROVER, L.A fast quantum mechanical algorithm for database search.In: **Proc. of the Twenty-Eighth Annual ACM Symp.on Theory of Computing**,p. 212–219, 1996.

HENKEL, M. Quantum computer simulation: New worldrecord on JUGENE, 2010. Available athttp://www.hpcwire.com/hpcwire/2010-06-28/quantum-computer simulation new world record on jugene.html(feb. 2013).

MARON, A.; REISER, R.; PILLA, M.; YAMIN, A.Quantum processes: A new approach forinterpretation of quantum gates in the VPE-qGMenvironment. In **Proceedings of WECIQ 2012**, p.80–89, 2012.

SHOR, P. Polynomial-time algoritms for primefactorization and discrete logarithms on a quantumcomputer. **SIAM Journal on Computing**, 1997.