SIMULAÇÕES QUÂNTICAS EM ARQUITETURAS PARALELAS VIA qGM$_C$-ANALYZER

MURILO FIGUEIREDO SCHMALFUSS[1]; ANDERSON BRAGA DE AVILA[1]; RENATA HAX SANDER REISER[1]; MAURÍCIO LIMA PILLA[1]

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

1. INTRODUÇÃO

A Computação Quântica (*CQ*) (NIELSEN; CHUANG, 2000) segue atingindo novos marcos rumo a construção de computadores quânticos. Apesar de todos os esforços, vários desafios técnicos limitam os sistemas atuais à apenas alguns *bits* quânticos (STEEB; HARDY, 2004). Devido à indisponibilidade de *hardware* quântico, o estudo e desenvolvimento de aplicações na *CQ* usualmente é feito estritamente pela especificação matemática das computações ou por meio de ferramentas de simulação. Este último caracteriza a abordagem mais prática, entretanto a complexidade computacional associada a simulação de sistemas quânticos a partir de computadores clássicos limita o tamanho dos sistemas que podem ser simulados.

O projeto no qual este trabalho está inserido busca consolidar a integração de vários esforços de pesquisa, com destaque para:

- Distribuição das computações em *cluster* (AVILA et al., 2012);
- Simulação quântica através de *GPUs* (MARON et al., 2013);
- Simulação quântica através de *CPUs multicore*.

Atualmente situado sob o contexto do ambiente de simulação quântica *VPE-qGM* (*Visual Programming Environment for the Quantum Geometric Machine Model*), este trabalho tem o objetivo de estabelecer o suporte à aceleração da biblioteca de execução do ambiente através de processadores *multicore*, beneficiando-se dos recursos providos pela biblioteca *OpenMP* (AYGUADE; CHAPMAN, 2003), uma interface para programação paralela multiplataforma baseada no modelo de execução *fork-join* para memórias compartilhadas. Consolida-se assim a biblioteca *qGM$_C$-Analyzer*, com a implementação, desenvolvimento e validação de algoritmo para simulação quântica em arquiteturas *multicore*, cuja modelagem foi introduzida em (SCHMALFUSS et al., 2012).

O ambiente *VPE-qGM*, fundamentado no modelo de processos *qGM* (*Quantum Geometric Machine Model*) (REISER; AMARAL, 2010), é constituído de construtores para modelagem e simulação gráfica de aplicações quânticas. De acordo com o modelo *qGM*, a noção de portas quânticas pode ser substituída pelo conceito de sincronização de processos elementares (*PEs*).

No ambiente *VPE-qGM*, o *PE* é um elemento estruturado por três atributos: (*i*) *Ação*: Corresponde às transformações quânticas aplicadas a diferentes *qubits* em um mesmo instante de tempo; (*ii*) *Parâmetros*: Contém dados auxiliares associados à definição das transformações quânticas; (*iii*) *Posição*: Posição de escrita em um espaço de memória global e compartilhada, na qual é armazenado o resultado calculado pelo *PE*.

Neste contexto, uma transformação quântica, aplicada à *N qubits*, pode ser modelada pela sincronização de 2^N *PEs*, cujas parametrizações satisfazem as condições equivalentes à definição dos vetores componentes da matriz (transformação unitária ou de medida) associada.

Assim, durante a simulação, ocorre a execução (sequencial ou síncrona) dos *PEs*, os quais têm suas correspondentes computações efetuadas pela biblioteca *qGM-Analyzer*, manipulando os dados presentes nas posições de memória e simulando o comportamento de um sistema quântico.

A biblioteca de execução dos *PEs*, denominada *qGM-Analyzer*, implementa otimizações que controlam o aumento exponencial dos vetores componentes das matrizes de definição do operador de múltiplos *qubits*, conforme introduzido em (MARON et al., 2011).

Os resultados relacionados comprovam a redução no consumo de memória durante a simulação, suportando algoritmos com 11 *qubits*. Entretanto, o tempo total de simulação obtido permanece elevado, devido a quantidade de operações necessárias para simular uma transformação quântica. Tendo em vista que o crescimento da complexidade algorítmica se dá de forma exponencial, abordagens paralelas e distribuídas foram estudadas para otimizar os cálculos envolvidos na simulação de uma transformação quântica.

2. METODOLOGIA

A implementação em *C++* manteve o mesmo algoritmo desenvolvido em (MARON et al., 2011), porém a forma de armazenamento dos dados e os algoritmos para acesso a esses dados foram modificados para diminuir a complexidade da biblioteca.

Os fatores que contribuíram para um melhor desempenho foram a utilização de vetores como alternativa as listas do *Python e* o fato de a linguagem *C++* ser compilada, permitindo que o compilador gere otimizações no código gerado.

As memórias de escrita e leitura são compartilhadas entre todos os *threads*, pois cada valor calculado é escrito em uma posição diferente da memória.

A divisão dos *threads* é feita de forma a manter juntos os valores das matrizes necessários para o cálculo de uma posição. Para isso as iterações são divididas levando em conta o número de colunas da primeira matriz envolvida na transformação. Nos casos em que a transformação possui apenas uma matriz, os *threads* são divididos de forma diferente, para que o trabalho seja distribuído de forma balanceada.

O controle do número de *threads* utilizadas pela biblioteca é definida por uma variável de ambiente (*OMP_NUM_THREADS*), gerando uma implementação mais flexível. Ou seja, dependendo da transformação quântica, tem-se um controle da granulosidade visando melhor desempenho da biblioteca.

3. TRABALHOS RELACIONADOS

Esta seção apresenta os trabalhos de pesquisa relacionados, mostrando as abordagens utilizadas por outras frentes de pesquisa. Existem vários trabalhos relacionados, dentre os quais se destacam o módulo *VirD-GM* (*Virtual Distributed Geometric Machine*) (AVILA et al., 2014) do ambiente *VPE-qGM*, a biblioteca *Libquantum* (BUTSCHER; WEIMER, 2003) e o simulador quântico *QuiDDPro* (VIAMONTES, 2007).

3.1 VirD-GM

A *VirD-GM* é um módulo do ambiente *VPE-qGM* responsável pelo gerenciamento da simulação distribuída. Ela é desenvolvida em *Java* e atualmente conta com suporte à simulação sequencial em *CPUs* e massivamente paralela através de *GPUs*, implementada utilizando a interface *JCUDA*.

3.2 Libquantum

A biblioteca *Libquantum* utiliza uma abordagem paralela utilizando *OpenMP* e otimiza a representação dos valores, omitindo os estados base cujas amplitudes são zero. Esta abordagem requer que os estados de base diferentes de zero sejam armazenadas assim como o número total de estados diferentes de zero.

3.3 QuiDDPro

O simulador *QuiDDPro* representa os estados quânticos multidimensionais e as transformações quânticas através de uma estrutura denominada *QuIDD* (*Quantum Information Decision Diagram*), definidos por blocos de valores. Os valores seguem padrões, sendo possível representá-los de forma compacta reduzindo os tempos de acesso à informação e consequentemente o tempo de simulação e o consumo de memória.

4. RESULTADOS E DISCUSSÃO

Para a realização dos testes foram simuladas transformações *Hadamards* de 14 a 21 *qubits*, abrangendo as quatro abordagens mencionadas no trabalho, todas determinísticas. O número de *qubits* utilizados nas simulações foi limitado pela expansão exponencial do espaço de memória. A medida de desempenho analisada foi o tempo médio de simulação, para cada caso de teste foram realizadas 15 simulações. A máquina utilizada para os testes possui as seguintes configurações: *Intel Core i7-3770, 8GiB RAM e GPU NVIDIA GT640*.

Tabela 1: Tempos de Simulação

Qubits	Tempo em segundos			
	qGMc-Analyzer	VirD-GM	Libquantum	QuiDDPro
14	1,1372	1,679	0,005067	0,029335
15	4,6993	2,083	0,006733	0,031468
16	14,779	2,319	0,008667	0,033068
17	59,78	7,469	0,016133	0,034135
18	245,16	23,604	0,033267	0,036802
19	1721,21	86,404	0,089133	0,037068
20	NE[1]	345,433	0,199667	0,038402
21	NE[1]	1364,844	0,420133	0,040002

[1]Não executado devido ao elevado tempo de simulação necessário

Os resultados obtidos podem ser observados na Tabela 1, as simulações executadas na *qGM$_C$-Analyzer* e na *VirD-GM* tiveram tempos de execução mais elevados que as outras abordagens, no entanto este resultado já era esperado. Devido a forma como a representação do vetor de estados é implementada no *QuIDDPro* e na *Libquantum* os tempos de acesso e o número de operações realizadas são muito menores. O desvio padrão das simulações foram omitidos da tabela por estarem todos abaixo de 1,6%.

Ainda nestes resultados é possível observar que os tempos de simulação para 14 *qubits* na *qGM$_C$-Analyzer* foram menores que na *VirD-GM*. Sendo a *VirD-GM* um gerenciador de simulação distribuído, a cópia de grande volume de dados para um número de operações não muito alto interfere no desempenho da simulação.

5. CONCLUSÕES

Com a abordagem distribuída da *VirD-GM* e a paralela da *qGM$_C$-Analyzer*, um modelo de computação heterogênea englobando as duas abordagens é justificada para otimizar os tempos de simulações. Transformações que necessitam um elevado número de operações seriam destinadas as *GPUs* enquanto as demais transformações seriam distribuídas entre as *CPUs*. O desafio desta proposta é o controle do volume de dados transportados para cada máquina do *cluster*, pois para cada nodo é necessário copiar toda a memória, ocasionando um grande fluxo de dados, limitando o desempenho.

Para trabalhos futuros busca-se uma otimização da estrutura que representa o vetor de estados, buscando padrões para a simplificação da estrutura.

6. REFERÊNCIAS BIBLIOGRÁFICAS

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

AVILA, A.; MARON, A.; REISER, R.; PILLA, M; YAMIN, A. **GPU-Aware distributed quantum simulation**. In: Proceedings of the SAC'14, 2014.

AYGUADE, E.; CHAPMAN, B. **Introduction: Special issue: OpenMP**. Scientific Programming, 2003.

BUTSCHER, B.; WEIMER, H. **Simulation Eines QuantencomputersB**. 2003.

MARON, A.; ÁVILA, A.; REISER, R.; PILLA, M. **Introduzindo uma nova abordagem para simulação quântica com baixa complexidade espacial**. In: Anais do DINCON 2011, 2011.

MARON, A.; REISER, R.; PILLA, M. **High-performance quantum computing simulation for the quantum geometric machine model**. In: Proceedings of CCGRID 2013, 2013.

NIELSEN, M.; CHUANG, I. **Quantum Computation and Quantum Information**. Cambridge University Press, 2000.

REISER, R.; AMARAL, R. **The quantum states space in the qGM model**. In: Proceedings of the III WECIQ, 2010.

SCHMALFUSS, M.; MARON, A.; REISER, R.; PILLA, M. **qGM$_C$-Analyzer: Biblioteca para suporte à simulação quântica em C++**. In: Proceedings of the XIII WSCAD-WIC, 2012.

STEEB, W.; HARDY, Y. **Problems and solutions in quantum computing and quantum information**. World Scientific, 2004.

VIAMONTES, G. **Efficient Quantum Circuit Simulation**. Phd thesis, The University of Michigan, 2007.