Simulando Passeios Quânticos em Processadores Vetoriais*

Félix D. P. Michels[1], Philippe O. A. Navaux[1], Paulo Motta[2], Renato Portugal[2]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

[2]Laboratório Nacional de Computação Científica
Petrópolis, RJ – Brasil

{felix.junior, navaux}@inf.ufrgs.br, prmottajr@gmail.com, portugal@lncc.br

Abstract. *Simulation is key in preparing quantum applications. One of those is Hiperwalk, a Quantum Walk simulator. Furthermore, vector accelerators arrive as hardware that can substitute GPUs in certain applications. This work focuses on adapting the established Hiperwalk simulator for the NEC SX-Aurora architecture, looking for performance improvements. We can then analyze the differences in implementation execution and analyze the performance of each of these architectures concerning the Hiperwalk simulator, achieving an increase of up to 75% in performance using a* 2^{15} *nonzero elements input matrix in SXAurora.*

Resumo. *Simuladores são fundamentais na preparação de aplicações quânticas. Um deles é o Hiperwalk, um simulador de passeios quânticos. Além disso, os aceleradores vetoriais chegam como hardware que pode substituir as GPUs em certas aplicações. Este trabalho se concentra na adaptação do simulador Hiperwalk estabelecido para a arquitetura NEC SX-Aurora, buscando melhorias de desempenho. Podemos então analisar as diferenças na execução da implementação e analisar o desempenho de cada uma dessas arquiteturas em relação ao simulador Hiperwalk, alcançando um aumento de até 75% no desempenho usando uma matriz de entrada* 2^{15} *elementos não nulos na SX-Aurora.*

1. Introdução

É inevitável que os avanços da computação quântica irão aumentar o poder e a eficiência computacional [Easttom 2021]. Computadores modernos necessitam dessas melhorias para processar sistemas biológicos complexos, estruturas químicas e novos materiais [Trabesinger 2017]. A IBM alcançou 127 qubits em seu mais recente chip de computação quântica, marcando um novo marco para toda a indústria, a primeira contagem de qubits de três dígitos [Ball 2021]. A IBM tem uma meta de ultrapassar 1000 qubits até 2023, visando um aumento nunca antes visto no poder computacional.

No entanto, o público não tem acesso completo a esses computadores e processadores quânticos. Os simuladores quânticos são a melhor alternativa para preencher essa

*Este trabalho foi parcialmente financiado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, pelo projeto Petrobras (2016/00133-9, 2018/00263-5) e pelo projeto "GREEN-CLOUD: Computação em Cloud com Computação Sustentável" (#16/2551-0000 488-9), da FAPERGS e do CNPq, programa PRONEX 12/2014 e projeto 406182/2021-3.

lacuna entre a comunidade científica em geral e o estado da arte na ciência da computação quântica. Uma categoria desses simuladores são os simuladores de caminhada quântica como o simulador Hiperwalk [Pedro C. S. Lara and 2017].

Arquiteturas distribuídas multi-core são uma alternativa atraente a ser explorada na simulação quântica de larga escala. O grande potencial dessas arquiteturas na escalabilidade é a exigência rigorosa de comunicação entre os núcleos quando os qubits precisam interagir. Para isso, escolhemos a arquitetura vetorial SX-Aurora TSUBASA da NEC para analisar seu desempenho com arquiteturas já estabelecidas.

Uma das principais características desses processadores vetoriais é a possibilidade de utilizar uma instrução para reproduzir centenas de operações. As arquiteturas vetoriais têm grande potencial para aplicações científicas altamente paralelizáveis. Entre essas aplicações estão aplicações numéricas, previsão de tempo, processamento multimídia [Kshemkalyani 2012], simulação de colisão [Hennessy and Patterson 2019], compressão de dados, entre outras.

Portanto, os principais objetivos deste artigo são (1) apresentar o simulador Hiperwalk, brevemente explicando seus casos de uso e (2) descrever a implementação do *kernel* do Hiperwalk na SX-Aurora TSUBASA da NEC, bem como (3) fazer um análise de nossa implementação comparando-a com as GPUs da NVIDIA. O principal problema para desenvolver um *kernel* para a arquitetura da NEC era sua incapacidade de compilar e executar código OpenCL. Assim, para rodar o simulador Hiperwalk na SX-Aurora, a biblioteca padrão do Neblina-core foi renovada.

2. Trabalhos Relacionados

O artigo *Simulation of Quantum Walks using HPC* de Pedro Lara, Aaron Leão e Renato Portugal descreve o simulador Hiperwalk utilizado neste trabalho [Pedro C. S. Lara and 2017]. Este trabalho é um dos poucos simuladores de caminhada quântica disponíveis ao público e um dos únicos a utilizar paralelismo para a simulação. Este simulador pode funcionar com arquiteturas vetoriais e *Single Instruction Multiple Data* (SIMD). No entanto, o simulador é baseado em linguagens e bibliotecas que restringem a portabilidade, OpenCL, sendo assim uma boa fonte para adaptação.

Semelhante a este trabalho, a publicação de Komatsu et al. mostra o potencial da arquitetura SX-Aurora TSUBASA. Uma comparação com outras arquiteturas, NVIDIA Tesla V100 e SX-Ace, entre outras, mostra resultados onde a SX-Aurora consegue rodar de forma eficiente, com desempenho de até 3,5×, além de obter um *speedup* maior de até 2,8×[Komatsu et al. 2018].

Trabalhos anteriores feitos por nosso grupo utilizaram a mesma arquitetura vetorial, SX-Aurora [Michels et al. 2020]. Michels discute em seu trabalho a análise de desempenho da arquitetura SX-Aurora utilizando um *benchmark* artificial e uma aplicação real de propagação de ondas, essencial para a indústria de petróleo e prospecção. Por meio de técnicas simples de otimização, como *loop unrolling* e *inlining*, foi possível obter melhorias de desempenho com a SX-Aurora de até $7,8\times$ com o *benchmark* NAS e até $1,9\times$ com o aplicativo da vida real.

3. Hiperwalk e Neblina-core

Hiperwalk é um programa de código aberto para gerar a dinâmica de modelos conhecidos de caminhada quântica (QW) em um grafo genérico usando HPC. Existem quatro modelos diferentes: Caminhada Quântica em Tempo Discreto (DTQW), Caminhada Quântica de Szegedy, Caminhada Quântica em Tempo Contínuo (CTQW) e Caminhada Quântica Escalonada. Nesta versão do simulador Hiperwalk, apenas dois desses modelos estão implementados atualmente: o modelo DTQW e o Staggered Quantum-Walk. Neste trabalho utiliza-se o modelo DTQW.

Para atingir seu objetivo, o Hiperwalk utiliza abstração de alto nível. Ele começa com um arquivo de texto simples como entrada. A biblioteca matemática Neblina-core lida com todos os cálculos. Primeiro, escolhemos nosso modelo de caminhada quântica dentro do arquivo de texto de entrada. Em seguida, o número de passos que estaremos tomando. O gráfico de entrada, com tipo e tamanho. Finalmente, precisamos do estado inicial do passeio quântico. Existem mais comandos opcionais e alguns são exclusivos para cada modelo de caminhada quântica.

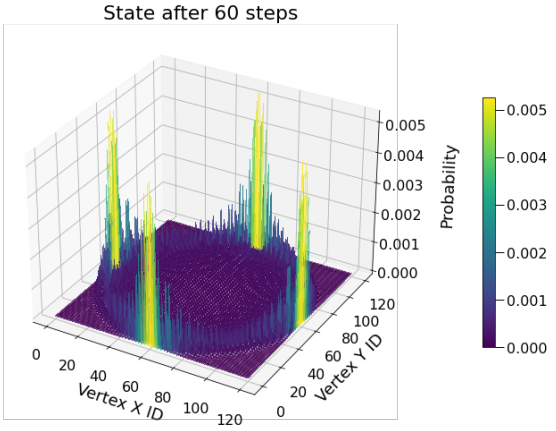

Figura 1. Distribuição de probabilidades de um passeio quântico em uma malha quadrada de 121 elementos.

Na Figura 1 temos a distribuição de probabilidade de um passeio quântico em uma malha de 121 x 121, após 60 passos. O eixo x e y fornece suas posições, e o eixo z fornece a probabilidade [%].

O Neblina-core é uma biblioteca matemática que requer do usuário pouco conhecimento acerca de programação paralela, sendo assim fácil de usar. No entanto, como dito anteriormente, o Neblina-core é implementado usando uma linguagem de codificação não compatível com a arquitetura da NEC. Para executar o Hiperwalk na arquitetura vetorial SX-Aurora, todas as seções de código que usam OpenCL precisam ser substituídas pela implementação da biblioteca matemática da NEC.

Um exemplo de uma função substituída é a multiplicação de matrizes esparsas. A implementação original dependia fortemente do OpenCL, que foi alterado para usar principalmente as funções da NEC. Usando a biblioteca matemática da NEC, foi possível usar suas rotinas *Sparse Basic Linear Algebra Subprograms* (SBLAS). A estrutura básica desta função segue: armazenamento de matrizes, inicialização do *handle*, solução, finalização.

4. Ambiente de execução e experimentos

Quatro conjuntos de dados são apresentados neste trabalho, aumentando gradualmente de tamanho. Esses conjuntos de dados são matrizes quadradas com quantidade x de elementos diferentes de zero. Cada matriz tem, respectivamente, 1024, 4096, 16384 e 32768 entidades diferentes de zero. Como o último conjunto de dados contém grandes quantidades de dados, ele é perfeito para testar a vantagem da largura de banda da memória SX-Aurora em relação à NVIDIA. Cada conjunto de dados foi executado 20 vezes e a média de cada resultado é apresentada na seção de resultados. Os dados resultantes registrados foram o tempo de execução em segundos, MFLOPs e taxa de erros da *cache* para L1 e LLC (*Last Level Cache*).

Os experimentos utilizaram o ambiente SX-Aurora através dos recursos e infraestrutura do Parque Computacional de Alto Desempenho (PCAD), http://gppd-hpc.inf.ufrgs.br, no INF/UFRGS. O ambiente possui memória global de oito núcleos e *cache* L3, cada núcleo com *cache* de memória L1 e L2, uma unidade de processamento escalar (SPU) e uma unidade de processamento vetorial (VPU), com cada VPU contendo *buffer* de carga, *buffer* de armazenamento, e 32 *pipelines* vetoriais paralelos (VPP) [NEC 2020].

A microarquitetura Intel Cascade Lake representa a arquitetura x86 com a GPU P100 da NVIDIA. O processador Intel Xeon Gold 6226 possui 22 núcleos operando em uma frequência entre 2,7 GHz e 3,7 GHz. Cada núcleo possui 32 KB de *cache* L1 de dados e instruções e um *cache* L2 privado de 1 MB. O L3 compartilhado entre todos os núcleos tem capacidade de 16,5 MB, e a máquina também possui 192 GB de memória DRAM. A NVIDIA P100 é uma GPU Pascal com 3584 núcleos CUDA [NVIDIA 2016].

As instruções para instalação do simulador estão disponíveis no link: `http://qubit.lncc.br/qwalk/`. Neblina-core pode ser encontrado em: `https://paulomotta.pro.br/wp/2021/05/01/pyneblina-and-neblina-core/`.

5. Resultados do Hiperwalk e a vetorização

O primeiro resultado obtido, em segundos são apresentados na Figura 2. Os tempos de execução das duas arquiteturas NEC e NVIDIA para os quatro tamanhos de entrada, 1024, 4096, 16384 e 32768, respectivamente, são 161,85s, 256,99s, 1401,21s e 5021,94s em relação a SX-Aurora e o desempenho da NVIDIA é 117,9s, 224,4s, 1820,1s e 8814,1s.

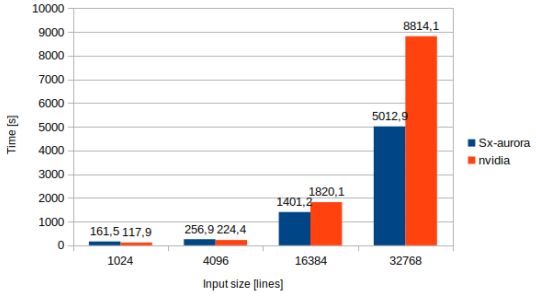

Figura 2. Tempo de execução para cada entrada de matriz (1024, 4096, 16384 e 32768)

Nota-se na Figura 2 que ao aumentar o número de entidades de entrada diferentes de zero, a arquitetura da NEC começa a ter um tempo de execução menor que a da NVIDIA. O caso de uso 16384 obteve uma redução média de aproximadamente 24% no tempo de execução, de 1820,1 segundos no P100 para 1401,2 segundos na Sx-Aurora. No conjunto de dados mais extenso houve um decréscimo de cerca de 43%, ganho de desempenho de cerca de 75%, atingindo aproximadamente 8814,1 e 5012,9 segundos, para o P100 e Sx-Aurora, respectivamente.

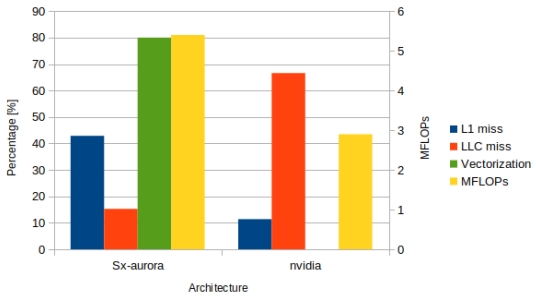

Figura 3. Vetorização [%], LLC e L1 taxa de erro da cache [%] e MFLOPs para a entrada de 32768 entidades não nulas

Agora, a última ilustração, Figura 3, demonstra alguns dos potenciais que a SX-Aurora da NEC tem nas circunstâncias certas. Devido a sua arquitetura vetorial, que pode realizar múltiplos cálculos com apenas uma instrução e, neste caso especificamente, a alta largura de banda de memória disponível.

Na Figura 2, o quarto conjunto de dados, com 32.768 entidades diferentes de zero, melhorou ainda mais o desempenho. Visualizando a Figura 3, temos ampla evidência. A vetorização na SX-Aurora conseguiu atingir cerca de 80%. Seu comprimento vetorial estava próximo aos limites da SX-Aurora, com 223 unidades de 256. A SX-Aurora teve apenas 15% de taxa de acertos de cache LLC e 43% para a proporção de acertos de cache L1. A NVIDIA novamente se saiu melhor para a taxa de acerto do LLC, com 67% e a taxa de acerto do cache L1 de 11%. Em relação ao desempenho bruto, a arquitetura SX-Aurora alcançou cerca de 5,51 MFLOPs e a arquitetura P100 cerca de 2,94 MFLOPS.

6. Conclusão e Trabalhos Futuros

A simulação de caminhada quântica é uma ferramenta essencial para estudar Computação Quântica. Portanto, seu desempenho deve estar perfeitamente sintonizado com a arquitetura desejada. É verdade, especialmente para o simulador Hiperwalk e arquiteturas vetoriais, devido à forma como o simulador é projetado, utilizando cálculos vetoriais e vetoriais matriciais. No entanto, a interface com a biblioteca Neblina-core, que é baseada em OpenCL para seus cálculos paralelos, dificulta a portabilidade do simulador.

Este trabalho utiliza uma arquitetura vetorial para tentar colher seu potencial em um software de caminhada quântica chamado Hiperwalk. A arquitetura escolhida é o acelerador vetorial SX-Aurora. Mostramos que é possível utilizar seus pontos fortes, melhorando o desempenho geral. No entanto, é necessário considerar as vantagens desse tipo de arquitetura e os pontos positivos específicos que a SX-Aurora possui.

Alcançamos um ganho de desempenho de cerca de 75% para o último caso, significando que tamanhos de entrada maiores favorecem a arquitetura da NEC. Por meio desses resultados, podemos atestar que, além da vantagem de maior largura de banda de memória, a vetorização automática fornecida pelo compilador da NEC é fundamental para seu sucesso. Nossa principal evidência é que o comprimento do vetor de vetorização aumenta a cada passo no tamanho da entrada. Embora a porcentagem geral de vetorização tenha diminuído, o comprimento do vetor mais considerável compensa essa perda e melhora ainda mais o desempenho.

Como trabalhos futuros, é possível aplicar diferentes técnicas de otimização para melhorar a vetorização e as taxas de acerto de cache. Essas técnicas variam de práticas gerais, como loop urolling e loop tiling, semelhante à implementação de Michels [Michels et al. 2020], até práticas específicas de arquitetura, como diretivas exclusivas da SX-Aurora para forçar a vetorização em determinadas seções. Também esperamos estender o desenvolvimento do Neblina-core, produzindo um novo módulo com CUDA para avaliar melhor as GPUs da NVIDIA.

Referências

Ball, P. (2021). First quantum computer to pack 100 qubits enters crowded race. *Nature*, 599(7886):542–542.

Easttom, W. (2021). *Quantum Computing and Cryptography*, pages 385–390. Springer International Publishing, Cham.

Hennessy, J. L. and Patterson, D. A. (2019). *Computer Architecture*. Cambridge: Horgan Kaufmann Publishers, Cambridge.

Komatsu, K., Momose, S., Isobe, Y., Watanabe, O., Musa, A., Yokokawa, M., Aoyama, T., Sato, M., and Kobayashi, H. (2018). Performance evaluation of a vector supercomputer sx-aurora tsubasa. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 685–696, Dallas Convention Center Arena. IEEE.

Kshemkalyani, P. A. (2012). Vector processors.

Michels, F., Serpa, M., Carastan-Santos, D., Schnorr, L., and Navaux, P. (2020). Otimização de aplicações paralelas em aceleradores vetoriais nec sx-aurora. In *Anais do XXI Simpósio em Sistemas Computacionais de Alto Desempenho*, pages 311–322, Porto Alegre, RS, Brasil. SBC.

NEC (2020). Sx-aurora tsubasa a100-1 series user's guide. `https://www.hpc.nec/documents/guide/pdfs/A100-1\_series\_users\_guide.pdf`. Accessed: 09/2021.

NVIDIA (2016). Nvidia tesla p100 gpu accelerator. `https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/tesla-p100/pdf/nvidia-tesla-p100-datasheet.pdf`. Accessed: 12-2021.

Pedro C. S. Lara and, Aaron Leão and, R. P. (2017). Simulation of quantum walks using HPC. *Journal of Computational Interdisciplinary Sciences*, 6:21.

Trabesinger, A. (2017). Quantum computing: towards reality.