Otimização de um algoritmo de simulação de transporte eletrônico de grafeno em GPU

Enrico C. A. Pereira[1], Arthur M. Passos[1], Calebe P. Bianchini[1,2]

[1]Faculdade de Computação e Informática – FCI
Universidade Presbiteriana Mackenzie – São Paulo, SP – Brasil

[2]CESAR – Centro de Estudos e Sistemas Avançados do Recife
Recife, PE – Brasil

enrico.pereira@mackenzista.com.br, arthurmoreirap@gmail.com,

calebe.bianchini@mackenzie.br, cpb@cesar.org.br

Abstract. *This work presents the optimization of an electronic transport simulator for twisted bilayer graphene, based on the Corbino geometry, by replacing the CPU-based PARDISO solver with the GPU-based cuDSS solver. The CUDA implementation preserves numerical accuracy and reduces the total execution time by 58.3%, achieving a 2.15× speedup and enabling more efficient conductance simulations for studies in condensed matter physics.*

Resumo. *Este trabalho otimiza um simulador de transporte eletrônico em grafeno bicamada torcido, baseado na geometria de Corbino, por meio da substituição do solucionador PARDISO em CPU pelo cuDSS em GPU. A implementação em CUDA preserva a precisão numérica e reduz o tempo total de execução em 58,3%, alcançando ganho de 2,15× e viabilizando simulações de condutância mais eficientes para estudos em física da matéria condensada.*

1. Introdução

O estudo de materiais bidimensionais tem ganhado destaque na física da matéria condensada, especialmente com o crescimento dos estudos sobre o grafeno e suas diversas configurações estruturais [Bahamon et al. 2013]. Entre essas configurações, o grafeno bicamada torcido (*Twisted Bilayer Graphene* – TBG) tem despertado grande interesse científico devido às suas propriedades físicas e eletrônicas peculiares, incluindo fenômenos associados à supercondutividade e a estados isolantes correlacionados, de acordo com [Nimbalkar and Kim 2020]. A investigação dessas propriedades, com ênfase na supercondutividade, depende fortemente de simulações que reproduzam o comportamento do transporte eletrônico no material. Um desses simuladores é o algoritmo de simulação de condutância elétrica baseado na geometria de Corbino desenvolvido por [Bahamon et al. 2013]. No entanto, a complexidade computacional envolvida nos algoritmos que realizam essas simulações representa um obstáculo significativo, tornando o processo lento, custoso e, muitas vezes, inviável para análises em larga escala ou de alta precisão.

O principal problema identificado é o alto custo computacional dos algoritmos utilizados para simular o transporte eletrônico no TBG com geometria de Corbino.

Atualmente, os cálculos são realizados utilizando a biblioteca Intel® MKL™, fazendo uso somente do processamento em CPU. No entanto, essa abordagem não aproveita o potencial de aceleração oferecido pelas GPUs, que poderiam melhorar o desempenho [Silva et al. 2022]. A ausência dessa otimização compromete a agilidade das simulações e limita o alcance das análises possíveis com os modelos atuais. Desta forma, delimita-se o problema na necessidade de reformulação e aceleração desses algoritmos para permitir simulações mais rápidas, precisas e acessíveis.

O objetivo deste trabalho é apresentar os resultados de uma otimização do algoritmo de simulação de condutância elétrica no grafeno bicamada torcido com geometria de Corbino utilizando processamento em GPUs. Dessa forma, espera-se promover um avanço computacional nas simulações de física e, consequentemente, facilitar os estudos e testes em materiais complexos como o TBG.

2. Referencial Teórico

A física da matéria condensada estuda propriedades físicas de sistemas microscópicos e estruturas de materiais sólidos, área que tem recebido atenção crescente com os materiais bidimensionais (2D), especialmente após a descoberta do grafeno em 2004 [Geim and Novoselov 2007]. Entre suas configurações estruturais mais relevantes está o grafeno bicamada torcido, no qual duas camadas de grafeno são sobrepostas com um pequeno ângulo de torção entre si, como o chamado "ângulo mágico" de aproximadamente 1,1°.

Para investigar fenômenos como a supercondutividade, são amplamente utilizadas simulações computacionais de transporte eletrônico, que calculam a condutância de elétrons sob diferentes geometrias e condições experimentais, como a geometria de Corbino [Bahamon et al. 2013]. Entretanto, essas simulações são computacionalmente intensivas, especialmente quando envolvem sistemas em larga escala.

O uso de GPUs pode acelerar simulações numéricas na física da matéria condensada devido à sua arquitetura paralela e ao suporte de bibliotecas como *cuBLAS*, *cuSPARSE*, *cuSOLVER* e *cuDSS*, que permitem a implementação eficiente de algoritmos científicos em GPU.

3. Desenvolvimento

Inicialmente, foi realizado um estudo do algoritmo original de simulação de condutância elétrica baseado na geometria de Corbino [Bahamon et al. 2013], visando compreender sua lógica, estrutura, funcionamento e operações matemáticas. Durante essa etapa, foram utilizadas ferramentas de medição de tempo de processamento, para identificar os *hotspots*, que são pontos de maior consumo computacional.

Uma vez identificados os *hotspots* e compreendido o algoritmo, foi realizada uma análise das bibliotecas mais adequadas e ferramentas para a otimização com GPUs. A ênfase principal foi em bibliotecas desenvolvidas pela NVIDIA, tais como o cuBLAS, cuSOLVER ou CuDSS.

Assim que definida qual a melhor biblioteca a ser utilizada para substituir trechos atuais do código, foram feitas as reimplementações e otimizações necessárias, visando a exploração dos cálculos na GPU.

Após a implementação completa do solucionador de sistemas lineares esparsos utilizando processamento em GPU, foram realizados testes de desempenho e coleta de tempos de execução para análise comparativa.

4. Experimentos e Resultados

Todos os experimentos foram conduzidos em um computador pessoal equipado com um processador Intel® Core i5-12400 com 40 GB de memória RAM, e uma GPU NVIDIA GeForce RTX 4060 com 8 GB de VRAM e 3072 CUDA *cores*, cuja capacidade de processamento em ponto flutuante de 64 bits (FP64) é de aproximadamente 236,2 GFLOPS. Para cada caso estudado, foram efetuadas 5 execuções, considerando-se posteriormente a média dos tempos obtidos para a construção das tabelas e gráficos apresentados nesta seção.

O ambiente de desenvolvimento consistiu no sistema operacional Ubuntu 24.04, utilizando o compilador `gcc` 13.3, além do *toolkit* CUDA 12 e compilação dos kernels em GPU via `nvcc` 12.

Durante a análise do algoritmo, foram realizados testes de tempo com todas as principais funções da implementação original, com o objetivo de encontrar possíveis gargalos. Dentre as funções analisadas, aquela que encapsula o uso da biblioteca PARDISO correspondeu a aproximadamente 82% do tempo total de execução.

A modificação desse código seguiu-se com a biblioteca cuDSS para sistema lineares esparsos de grande dimensão [Pereira 2025]. Durante a implementação do solucionador usando esta biblioteca, enfrentou-se diversos desafios. Entre eles, destaca-se a necessidade de conversão adequada dos tipos de dados, já que o formato exigido pelo cuDSS difere daquele utilizado no código original, como a conversão `MKLComplex` para `cuComplex`. Outro ponto crítico foi a definição do tipo de matriz, uma vez que o algoritmo do simulador não fornecia indicação explícita sobre sua estrutura, exigindo investigação adicional para garantir a compatibilidade com o método adotado.

Após os experimentos, as divergências entre os resultados originais e a versão para GPU tornam-se perceptíveis apenas a partir da 11^a casa decimal nos valores analisados da simulação. Essa diferença numérica decorre das particularidades dos algoritmos empregados e da representação em ponto flutuante conforme o padrão IEEE 754. As médias do tempo total de execuções das simulações para uma entrada específica do problema TBG com 2500 e 3500 iterações foram, respectivamente: (i) PARDISO: 177,48s e 247,73s; (ii) cuDSS: 73,72s e 103,42s. Esses resultados indicam que o uso de GPU reduziu em aproximadamente 58,3% o tempo total de execução.

Foram coletadas métricas de memória e desempenho, incluindo o consumo de VRAM para o cuDSS e RAM para o PARDISO, além do percentual de uso de GPU nas versões cuDSS. O cuDSS apresentou consumo médio de 522 MB de VRAM e manteve cerca de 1.536 CUDA *cores* ativos, enquanto o PARDISO utilizou 4382 MB de RAM. Os dados de consumo de GPU são apresentados na Figura 1.

5. Considerações Finais

Esse trabalho demonstra que a utilização de unidades de processamento gráfico (GPUs) constitui uma estratégia eficaz para aceleração de algoritmos que

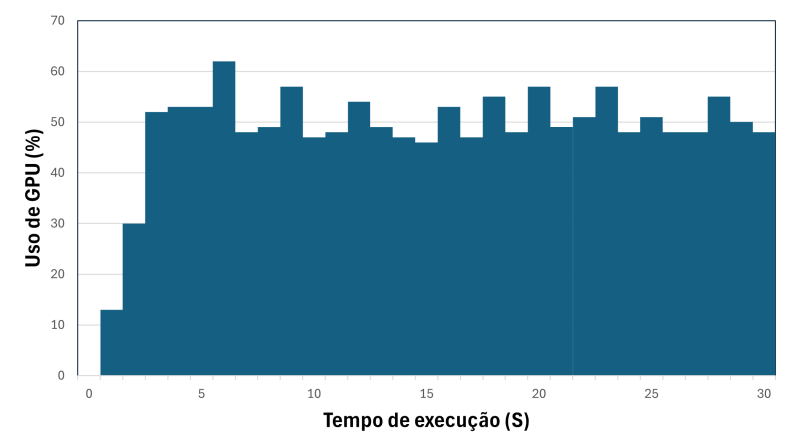

Figure 1. Percentual de utilização da GPU ao longo da execução da versão com cuDSS.

envolvem sistemas lineares de grande porte. A implementação apresentou ganho de desempenho de 2,15×, com redução de 58,3% no tempo de execução.

Portanto, entende-se que o uso de GPUs para simulações de condutância elétrica do grafeno, não apenas oferece ganhos de eficiência, como também é um passo importante nas pesquisas físicas do material, viabilizando que estudos sejam conduzidos com maior acessibilidade e menor custo computacional em um menor espaço de tempo.

Agradecimentos

Os autores agradecem o apoio da MackCloud, Laboratório Multidisciplinar de Computação Científica e Nuvem[1]; e do projeto SPRACE – Processo nº 2018/25225-9, Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP). Este trabalho foi financiado em parte pelo Fundo Mackenzie de Pesquisa e Inovação (MackPesquisa) – Projetos nº 231009 e 251005.

References

Bahamon, D. A., Neto, A. H. C., and Pereira, V. M. (2013). Effective contact model for geometry-independent conductance calculations in graphene. *Phys. Rev. B*, 12:235433.

Geim, A. K. and Novoselov, K. S. (2007). The rise of graphene. *Nature Materials*, 6:183–191.

Nimbalkar, A. and Kim, H. (2020). Opportunities and challenges in twisted bilayer graphene: A review. *Nano-Micro Letters*, 12:20.

Pereira, E. C. A. (2025). Otimização para gpus do algoritmo de simulação de transporte eletrônico de grafeno. Trabalho de Conclusão: Ciência da Computação.

Silva, G. P., Bianchini, C. P., and Costa, E. B. (2022). *Programação Paralela e Distribuída com MPI, OpenMP e OpenACC para computação de alto desempenho*. Casa Do Codigo.

[1] `https://mackcloud.mackenzie.br`