UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

Gabriel Leoni Duarte

Técnicas de paralelismo com GPUs para análise de variantes virais

São José do Rio Preto
2025

Gabriel Leoni Duarte

Técnicas de paralelismo com GPUs para análise de variantes virais

Trabalho de Conclusão de Curso (TCC) apresentado como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação, junto ao Conselho de Curso de Bacharelado em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de São José do Rio Preto.

Financiadora: PIBIC

Orientador:

Prof. Dr. Geraldo Francisco Donegá Zafalon

São José do Rio Preto
2025

D812t

Duarte, Gabriel Leoni
Técnicas de paralelismo com GPUs para análise de variantes virais / Gabriel Leoni Duarte. -- São José do Rio Preto, 2025
57 p.

Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) - Universidade Estadual Paulista (UNESP), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto
Orientador: Geraldo Francisco Donegá Zafalon

1. Unidades de processamento grafico. 2. Polimorfismo de nucleotídeo único. 3. Bioinformática. 4. Alinhamento de sequências (Bioinformática). I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

Gabriel Leoni Duarte

Técnicas de paralelismo com GPUs para análise de variantes virais

Trabalho de Conclusão de Curso (TCC) apresentado como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação, junto ao Conselho de Curso de Bacharelado em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de São José do Rio Preto.

Financiadora: CNPq

Comissão Examinadora:

Prof. Dr. Geraldo Francisco Donegá Zafalon
UNESP – Câmpus de São José do Rio Preto
Orientador

Prof. Dr. Carlos Roberto Valêncio
UNESP – Câmpus de São José do Rio Preto

Prof. Dr. Diego Renan Bruno
UNESP – Câmpus de São José do Rio Preto

São José do Rio Preto
2025

"Look up, face forward, toward your chosen horizon... And just... walk on."

- Noah (Xenoblade Chronicles)

Agradecimentos

Agradeço primeiramente aos meus pais, Vicente e Karine, que sempre me incentivaram a continuar, a buscar mais, iluminando o meu caminho para seguir em frente; a motivação da minha irmã, Letícia, que se inspira em mim, me fortificando para ser um irmão melhor; e a todos os meus outros parentes que estiveram presentes na minha vida.

Agradeço aos meus orientadores, Geraldo e Vitória, por acreditarem no meu potencial, me proporcionando um laboratório, um projeto para me dedicar, e auxílio dentro e fora dele. Obrigado por me proporcionarem todas as risadas, todas as lições de vida, muito dos seus conhecimentos sobre a faculdade e computação.

Agradeço aos meus principais amigos da faculdade, Ana, Bruno, Guilherme, João, Luan, Tulio, Vinicius, William, sem vocês, todo esse percurso seria impossível de ser superado, agradeço por todos os trabalhos em grupo, conversas, reuniões, e principalmente pela oportunidade de ser chamado de amigo. Agradeço também a todos os outros amigos que fiz ao longo da jornada da faculdade, veteranos, colegas de sala, calouros, e vários outros que conheci em outras ocasiões.

Agradeço também aos meus amigos pessoais, Diogo, Eliam, Engos, Lucas, Kaiky, Roberto, Victor, aqueles que me ajudaram a descontrair da faculdade e me ensinaram a ser uma pessoa melhor, me ensinaram o valor da diversão.

Agradeço a UNESP do câmpus de São José do Rio Preto, e a todos os professores do departamento de computação e professores substitutos que fizeram parte da minha formação e agregaram muito a minha formação com seu conhecimento.

Agradeço, por fim, à Programa Institucional de Bolsas de Iniciação Científica (PIBIC), pelo apoio financeiro a este trabalho.

Resumo

A bioinformática é uma área emergente de extrema relevância, pois permite aos biólogos a utilização de ferramentas computacionais para análise mais eficiente de dados genômicos inseridos em diferentes contextos. Dentre estas principais tarefas a serem realizadas, tem-se a análise de nucleotídeos de polimorfismo simples (SNPs), e a análise de variantes virais. Diversos trabalhos têm sido realizados, porém, com o advento do sequenciamento de nova geração (NGS) e a geração de milhares de dados genômicos, uma análise manual torna-se inviável nos dias atuais com esse grande volume de dados. Com isso, faz-se necessário a utilização de métodos computacionais que se baseiam em computação paralela, feitos para o processamento de dados em larga escala. Com isso, este projeto tem como objetivo modelar e implementar métodos de análise de SNPs e variantes virais em dados genômicos de larga escada, utilizando Unidades de Processamento Gráfico (GPU) para a aplicação do paralelismo. Os resultados obtidos demonstraram que a implementação em GPU apresentou um desempenho computacional superior quando comparada à execução em CPU, reduzindo significativamente o tempo de processamento. Graças ao ganho computacional proporcionado pela utilização da GPU, será possível realizar importantes análises em uma estação de trabalho local, sem a necessidade de estruturas mais complexas, como *grids* e *clusters*.

Palavras-chave: Programação em GPU, Análise de SNPs, Redes One-Step, Bioinformática.

Abstract

Bioinformatics is an emerging area of extreme relevance, as it allows biologists to use computational tools to analyze genomic data in different contexts more efficiently. Among the main tasks to be carried out are the analysis of single nucleotide polymorphisms (SNPs) and the analysis of viral variants. Several studies have been carried out, but with the advent of next-generation sequencing (NGS) and the generation of thousands of genomic data, manual analysis becomes unfeasible nowadays with this large volume of data. This makes it necessary to use computational methods based on parallel computing, designed for large-scale data processing. With this in mind, this project aims to model and implement methods for analyzing SNPs and viral variants in large-scale genomic data, using Graphics Processing Units (GPU) to apply parallelism. The results obtained demonstrated that GPU implementation showed superior computational performance when compared to CPU execution, significantly reducing processing time. Thanks to the computational gain provided by the use of the GPU, it will be possible to carry out important analyses on a local workstation, without the need for more complex structures such as *grids* and *clusters*.

Keywords: GPU Programming; SNP Analysis; One-Step Networks; Bioinformatics.

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

AMS - Alinhamento Múltiplo de Sequências

AVX - Advanced Vector Extensions

CDC - Centers for Disease Control and Prevention

COVID-19 - Coronavirus Disease 2019

CPU - Central Processing Unit

Cryo-EM - Cryogenic Electron Microscopy

CUDA - Compute Unified Device Architecture

DNA - Deoxyribonucleic Acid

GPU - Graphics Processing Unit

Indel - Insertion and Deletion

JIT - Just-In-Time

NGS - Next-Generation Sequencing

RAM - Random Access Memory

RNA - Ribonucleic Acid

SARS-CoV-2 - Severe Acute Respiratory Syndrome Coronavirus 2

SIMD - Single Instruction, Multiple Data

SIMT - Single Instruction, Multiple Threads

SM - Streaming Multiprocessor

SNP - Single Nucleotide Polymorphism

SOLiD - Sequencing by Oligonucleotide Ligation and Detection

SPM - Sequências Por Milissegundo

VOC - Variants of Concern

VRAM - Video Random Access Memory

WSL - Windows Subsystem for Linux

Sumário

Capítulo 1

Introdução ao trabalho

1.1 Introdução

O avanço das pesquisas no campo da biologia permitiu uma evolução significativa, tanto nos aspectos computacionais quanto científicos. Uma das áreas em que se notou esse avanço foi na genômica, graças à contribuição de Watson e Crick com a descoberta da estrutura do DNA humano em 1953 (PROSDóCIMI et al., 2023). Posteriormente, novas descobertas foram realizadas, incluindo código genético, função dos ácidos nucleicos para proteínas, novos métodos para o sequenciamento das biossequências, entre outras.

As tecnologias utilizadas para o sequenciamento evoluíram de maneira significativa. Inicialmente, utilizava-se o método de Sanger (SANTOS et al., 2013), mas, a partir de 2005, surgiram as tecnologias NGS, ou Sequenciamento de Nova Geração (CARVALHO; SILVA, 2010). O principal intuito é a geração de pares de bases de maneira altamente eficiente, especialmente por conta da clonagem *in vitro*. Como resultado da utilização constante destas novas tecnologias, houve a geração massiva de dados, superando a capacidade que as técnicas experimentais tradicionais conseguiam suportar. Nesse cenário, a bioinformática surge se destacando como um dos recursos para superar essas barreiras e promover a análise e apresentação desses dados biológi-

cos (ARAÚJO et al., 2008).

Na última década, a bioinformática consolidou-se como um dos pilares no desenvolvimento de pesquisas na área de genômica (KANEHISA; BORK, 2003). Acompanhando o crescimento explosivo de dados obtidos por meio de pesquisas experimentais (LIU; ABBAS, 2006), essa área impulsionou o avanço de diversos campos, desde análise de sequências proteicas e alinhamento de biossequências até o desenvolvimento de novos medicamentos.

Um dos principais desafios da bioinformática tem sido o processamento dessa grande quantidade de dados para gerar informações relevantes às pesquisas biológicas, que consistem fundamentalmente na comparação de várias biossequências (FIGUEIRÔA et al., 2015). Nesse contexto, a aplicação de técnicas de paralelismo com GPU surgem como uma solução para lidar com essa grande quantidade de dados.

Para superar as limitações impostas pela volumetria de dados, surgem os métodos computacionais de Needleman-Wunsch (NEEDLEMAN; WUNSCH, 1970) e Smith-Waterman (SMITH; WATERMAN, 1981) para alinhamento de pares de sequências. O primeiro, com uma abordagem de análise de alinhamento global e o segundo com vistas a alinhamento local. Ambos objetivam encontrar similaridades entre sequências genômicas, de modo a tentar determinar a homologia entre as proteínas e sequências de DNA (NEEDLEMAN; WUNSCH, 1970). A seguir, na figura 1.1, encontra-se um exemplo de alinhamento par-a-par de sequência.

Figura 1.1: Alinhamento de Sequências simples.
Fonte: (PROSDóCIMI et al., 2023)

Com o aumento do número de sequências genômicas gerados pelos sequenciadores

automáticos, a análise par-a-par tornou-se impraticável, necessitando-se, dessa forma, do desenvolvimento de algoritmos para alinhamento de múltiplas sequências. Estes, também conhecidos como Alinhamentos Múltiplos de Sequências (AMS), são algoritmos probabilísticos, ou heurísticos, que na sua concepção permitem trabalhar com um conjunto maior do que duas sequências. Na figura 1.2, encontra-se um exemplo de um alinhamento múltiplo realizado com alguns pares de sequências.

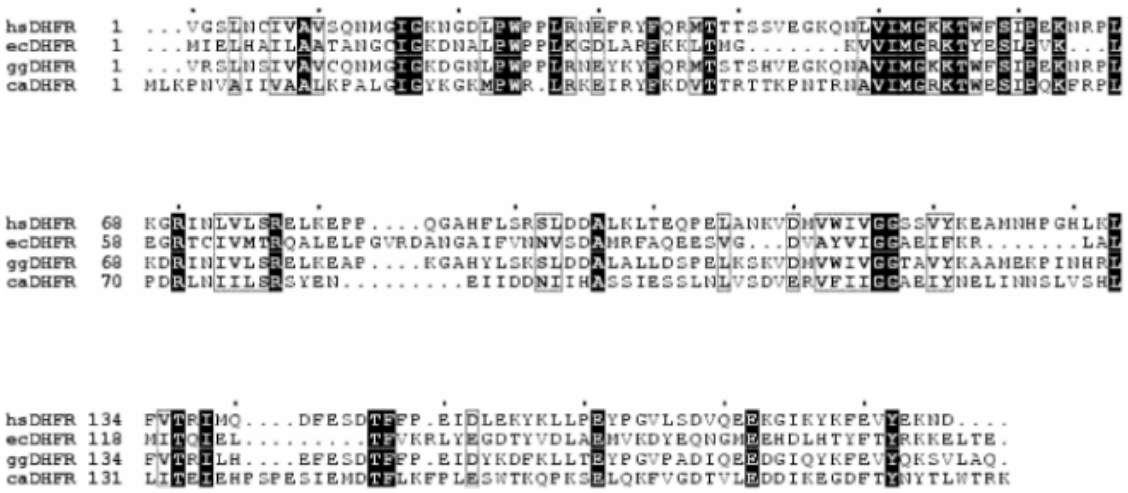

Figura 1.2: Alinhamento de sequências feito pelo programa CLUSTAL.
Fonte: (SANTOS et al., 2023)

Os métodos computacionais citados anteriormente possibilitaram um avanço na área de reconhecimento de padrões no campo da bioinformática, pois anteriormente todo o processo para identificar o alinhamento de sequências era realizado manualmente, sendo possível iniciar o alinhamento de sequências de maneira automatizada e entender padrões entre duas ou mais biossequências mais facilmente (LEMOS; CASANOVA, 2023). O alinhamento consiste em igualar o comprimento das sequências de entrada por meio da inserção de *gaps* (representados pelo caractere -") entre as bases para maximizar e ressaltar suas semelhanças, avaliando as combinações produzidas através de matrizes de substituição para aferir a qualidade biológica do resultado. Porém, uma vez que as possíveis combinações de inserção de *gaps* são muito grandes, cada uma gerando um alinhamento diferente, é necessária a utilização de algoritmos

que consigam varrer o espaço de busca visando um alinhamento ótimo, uma vez que não há garantia de encontrar o melhor alinhamento (WANG; JIANG, 1994).

Com o advento do sequenciamento de nova geração (NGS) e a massiva quantidade de dados, os vírus foram um dos focos principais nos últimos tempos, principalmente com os impactos causados pela pandemia de COVID-19, doença causada pelo vírus SARS-CoV-2, enfrentada pela sociedade recentemente. Na figura 1.3 é possível observar diferentes variantes do vírus.

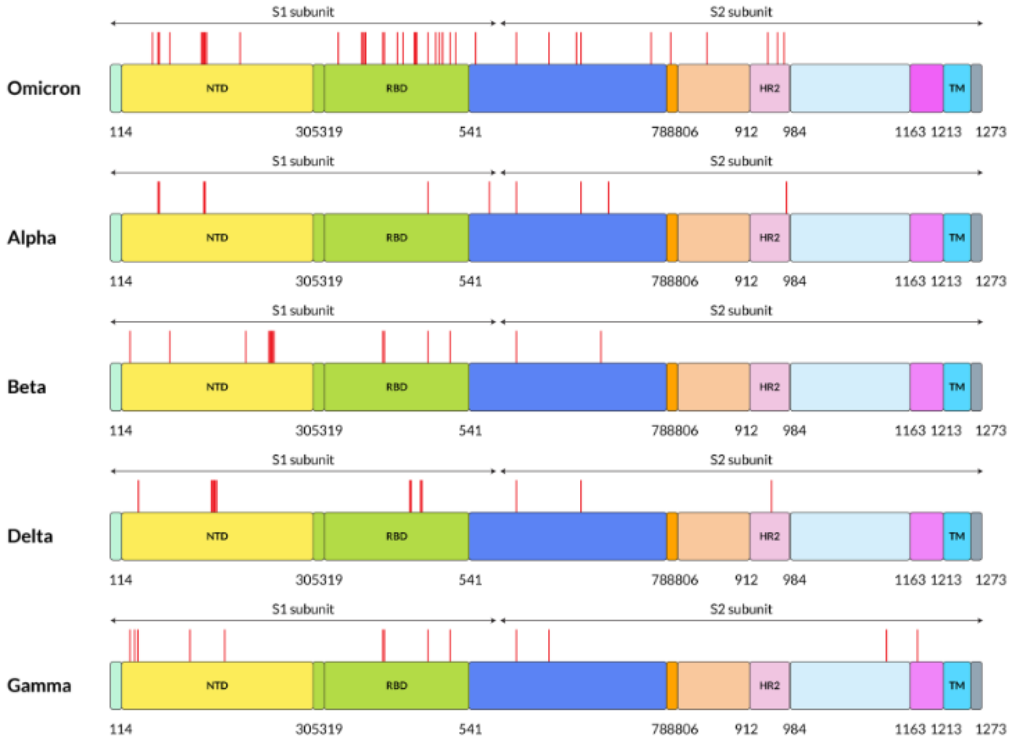

Figura 1.3: Variantes Virais do SARS-CoV-2.
Fonte: (ARAF et al., 2022)

Entender as diferenças biológicas entre o vírus original e suas variantes permite que sejam identificadas individualidades presentes em seu código genético. O estudo destes organismos é complexo e dificultado por conta das suas dimensões microscópicas (KARAS; HERMEL; GÜLLICH, 2018). Para facilitar tal estudo, faz-se necessária a utilização da bioinformática e seus recursos.

Ao tornar mais eficiente a aplicação da bioinformática nas variantes virais, as

contribuições na área médica são ilimitadas, pois avanços em áreas de manipulação e terapia gênica permitem o tratamento dos genes e seus mecanismos regulatórios (ARAÚJO et al., 2008). Além disso, os custos necessários para execução de tarefas como sequenciamento de DNA são drasticamente reduzidos conforme o avanço das tecnologias, tornando essas análises menos custosas a longo prazo. Na Figura 1.4, é possível observar o crescimento exponencial da quantidade de bases que podem ser sequenciadas a um custo fixo, evidenciando o aumento da eficiência dessas tecnologias ao longo dos anos.

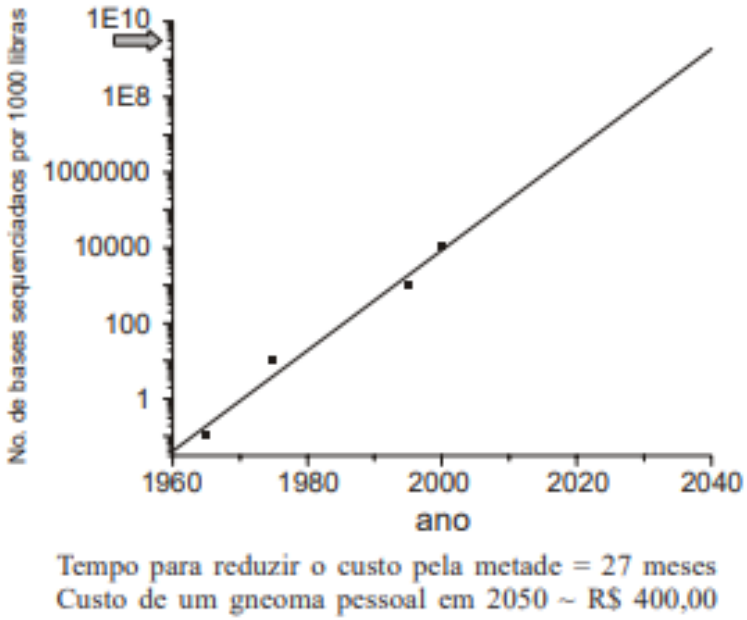

Figura 1.4: Quantidade de bases sequenciadas a um custo de 100 libras ao longo dos anos.
Fonte: (OLIVA, 2008)

1.2 Objetivos

O projeto tem como objetivo a otimização do tempo de execução de um método computacional que busca realizar o cálculo da distância de Hamming entre as várias biossequências. Após esse cálculo das distâncias, será realizada uma Detecção da Igualdade das Sequências, buscando entender quais as semelhanças entre elas. Por fim,

ao compreender as igualdades, é possível notar os traços de Variação Genética dentro delas, portanto, todo o método é focado na construção das relações entre os vírus e suas determinadas variantes virais, para possibilitar uma análise por meio da aplicação em Redes *One-Step*. Para tal otimização, serão empregadas técnicas de paralelismo por meio da utilização da GPU, tendo em vista a vantagem apresentada anteriormente em comparação à CPU.

Após esta implementação, busca-se realizar uma análise do desempenho na execução das redes em GPU, comparando o mesmo método em ambientes distintos para medir o crescimento de desempenho ao aplicar tais técnicas. Como o projeto tem como propósito auxiliar o campo da medicina e biologia a compreender a evolução de certos indivíduos ao decorrer dos anos, por meio de uma análise minuciosa das ligações dentro da rede, tal atividade deve ser feita de maneira mais eficiente possível.

1.3 Justificativa

A hipótese de que a implementação de algoritmos de bioinformática em Unidades de Processamento Gráfico (GPUs) pode drasticamente acelerar a análise de variantes virais se justifica pelo sucesso já consolidado da computação de alto desempenho na área. Um dos estudos que fundamenta esta abordagem é o de Schmidt et al. (2024) com o CUDASW++ 4.0, que implementou um *software* para análise de sequências de proteínas na arquitetura CUDA da NVIDIA (SCHMIDT et al., 2024). O núcleo do projeto utiliza o algoritmo de Smith-Waterman e, apesar de o algoritmo ser extremamente sensível para análise, sua complexidade quadrática o torna computacionalmente caro se comparado a outros, tal como visto na figura 1.5:

Algoritmos	Tempo Execução (Fig. 11)	Tempo Execução (Fig. 12)	Performance	Localizações de Memória	Notação Big O
FDASA	~0,163 ms	~0,053 ms	Alta	O(3M+1) se duas sequência tem o mesmo comprimento O(3M+2) se a sequência tem comprimento diferente	O(3M+1) se duas sequência tem o mesmo comprimento O(3M+2) se a sequência tem comprimento diferente
Needleman-Wunsch algorithm	~1,273 ms	~0,140 ms	Baixa	O(M*N) conforme precisamos para preencher toda a matriz	O(M*N) conforme precisamos para preencher toda a matriz
Smith-Waterman	~1,042 ms	~0,139 ms	Baixa	O(M*N) conforme precisamos para preencher toda a matriz	O(M*N) conforme precisamos para preencher toda a matriz

Figura 1.5: Análise de Desempenho algoritmo Smith-Waterman.
Fonte: Adaptado de (SHEHAB; KESHK; MAHGOUB, 2012)

Apesar deste alto custo, utilizando técnicas de programação em GPUs e comunicação eficiente entre *threads*, foram obtidos resultados quase 5 vezes maiores com o uso de GPU, mesmo se comparado a uma CPU de altíssimo desempenho, demonstrando que mesmo para um algoritmo lento, a implementação em GPU o torna significativamente mais rápido que em CPU.

No que se refere à aplicação específica em genômica viral, a escolha se justifica pela urgência e pelo impacto direto na saúde pública. Um dos principais adventos que evidenciou essa importância foi a pandemia do SARS-CoV-2, com mais de 30 milhões de casos e 600 mil óbitos somente no Brasil (PERRONE, 2022). A vigilância epidemiológica de patógenos depende da capacidade de processar um volume imenso de amostras rapidamente para rastrear a disseminação de novas variantes de preocupação (VOCs). Conforme destacado por Gardy e Loman (2018), os gargalos computacionais representam um dos maiores desafios na aplicação da genômica para o monitoramento de surtos em tempo real (GARDY; LOMAN, 2018). Diferentemente da análise de genomas eucarióticos complexos, o fluxo de trabalho para genomas virais é mais direto, e

a etapa de alinhamento consome a maior parte do tempo computacional. Portanto, otimizar este passo com GPUs significa tornar a análise genômica uma ferramenta viável não apenas para pesquisa, mas para a tomada de decisão clínica e sanitária imediata.

Também vale a pena pontuar que a aceleração por GPU não é uma estratégia isolada apenas para análise viral, mas sim uma tendência consolidada em vários domínios da biologia computacional. No campo da inteligência artificial aplicada à biologia, exemplificado pelo AlphaFold da DeepMind, demonstra que o treinamento das redes neurais profundas torna essencial o uso de GPUs para realização do treinamento (JUMPER et al., 2021). Outro exemplo é no campo de dinâmica molecular, com grande parte dos algoritmos já desenvolvidos em CPU. Graças a novos avanços, como o AMBER, GROMACS e NAMD, os tempos de execução de uma simulação se aproximam cada vez mais do tempo de experimentação em laboratório (PANDEY et al., 2022). Na biologia estrutural, a análise de imagens de crio-microscopia eletrônica (Cryo-EM) foi revolucionada por softwares como o RELION, que dependem de GPUs para reconstruir estruturas atômicas de macromoléculas (ZIVANOV et al., 2018). A aplicação de GPUs no presente trabalho, portanto, insere-se neste amplo contexto de inovação computacional, validando a abordagem como uma fronteira relevante e promissora para a solução de problemas complexos em bioinformática.

1.4 Motivação

A análise de variantes virais em tempo real é uma das tarefas de maior impacto na ciência moderna, servindo como a principal ferramenta da vigilância genômica e da epidemiologia molecular. Sua importância vai além da pesquisa acadêmica, funcionando como um pilar para a segurança da saúde global. O desenvolvimento de ferramentas computacionais mais rápidas e escaláveis para essa análise é, portanto, diretamente motivado por suas aplicações críticas, como em Monitoramento de Pandemias e Surtos (GARDY; LOMAN, 2018), Desenvolvimento de Vacinas e Terapias (RO-

DRIGUES, 2024), Diagnóstico Clínico e Medicina de Precisão (NETO et al., 2023) e Compreensão da Biologia Viral e Evolução (SOUZA, 2022). Melhorar as ferramentas para a análise de variantes auxilia diretamente na obtenção de resultados clínicos confiáveis e em um tempo útil, podendo salvar várias vidas. Portanto, tornar essa tecnologia acessível e viável para laboratórios de saúde, hospitais e centros de pesquisa em todo o mundo é a principal motivação deste trabalho.

1.5 Metodologia

Com os grandes avanços na biologia e o crescimento da quantidade de dados genômicos, a metodologia utilizada no desenvolvimento deste projeto tem como fundamentação um levantamento bibliográfico baseado no tema de aplicação de métodos e estratégias computacionais para análise de variantes virais, de modo a conseguir um melhor desempenho e aplicar o que há de mais recente como tecnologia.

Para a realização da análise das variantes de maneira mais detalhada e concisa, foram utilizadas Redes *One-Step*. No cálculo da rede, houve uma conexão de duas variantes com apenas a diferença de uma base da biossequência, tornando a visualização mais otimizada para uma análise profunda. Além desta visualização otimizada, tem-se uma análise de $n(n-1)/2$ quantidade de pares possíveis.

Para a análise de várias biossequências e suas diferenças de bases, a GPU foi utilizada, uma vez que esta técnica facilita a aplicação de paralelismo, ideal para a construção das Redes *One-Step* e as comparações entre as diferentes sequências.

Após esta comparação, uma série de dados disponibilizados pelo Centros de Controle e Prevenção de Doenças (CDC) foram introduzidos para uma análise minuciosa. Foram executados testes e o tempo de execução com quantidades variadas de sequências, medindo-os e armazenando-os. Ao final, após uma análise dos tempos medidos previamente, uma avaliação de significância biológica foi aplicada, buscando compreender qual a variação aplicada e como isso afeta o vírus.

Após encontrar o alinhamento ótimo, tem-se, em teoria, a melhor biossequência formada pelos diversos alinhamentos diferentes. Com isto, agora é necessário encontrar a diferença entre a sequência original e sua variante, mas sem conhecimento da sequência que originou esta variante gerada pelo algoritmo. Para isto, então, utiliza-se as redes *One-Step*. O método *One-Step* consiste em conectar sequências estreitamente relacionadas sem qualquer suposição sobre sua origem ancestral (ANDRADE, 2020), apresentando visualmente uma relação de hereditariedade entre as várias sequências analisadas. Na Figura 1.6, é possível observar a estrutura resultante dessa abordagem, onde os nós representam as sequências e as arestas indicam a proximidade genética.

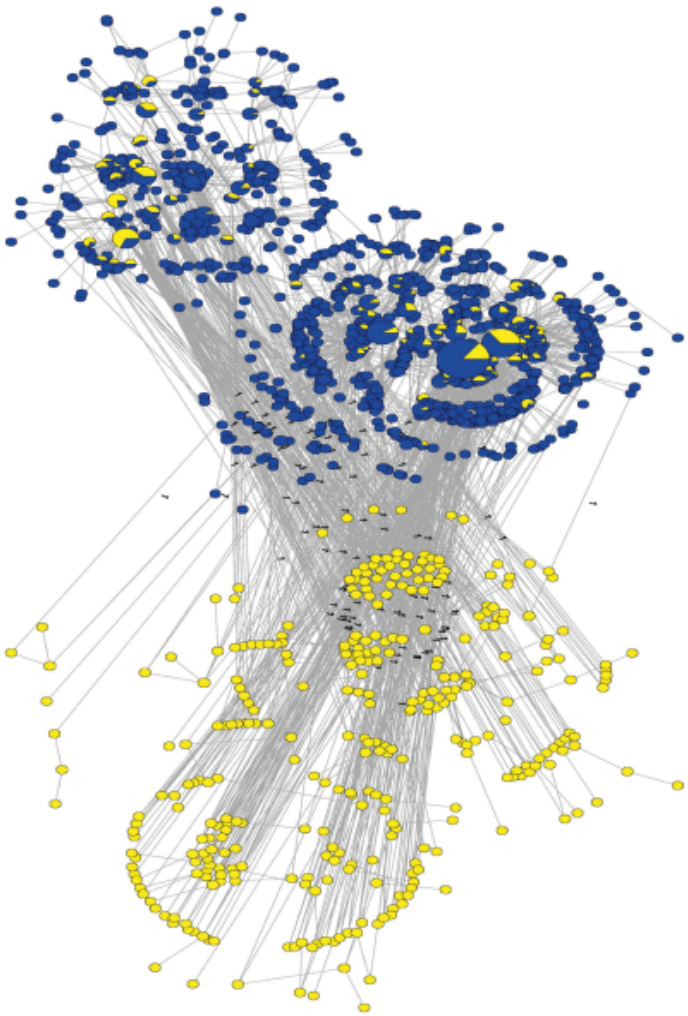

Figura 1.6: Apresentação da Rede *One-Step*.
Fonte: (ANDRADE, 2020)

A construção de redes *One-Step* é fundamental para a análise das variantes virais, tendo em vista que esta realiza a conexão entre as diferentes biossequências, possibilitando identificar pontos de alteração e com propensão à mutação, apontando a origem daquela nova característica da variante (CAETANO et al., 2016).

Para uma execução mais eficiente das análises que foram realizadas durante o projeto, tal qual a construção das redes *One-Step*, demanda-se grande poder de processamento devido à grande quantidade de sequências a serem analisadas. Por muito tempo, esse processo foi exclusivamente realizado pela Unidade Central de Processamento (CPU) por ser a única unidade para este processamento dentro das máquinas. Porém, com o advento das Unidades de Processamento Gráfico (GPUs), tornou-se possível a execução de várias tarefas simultaneamente, também chamado de paralelismo, com maior eficiência (DOMINGOS,). Mesmo com a introdução de multiprocessadores (processadores com múltiplos núcleos), o desempenho das GPUs se provou superior. Na Figura 1.7 é possível observar as diferenças arquiteturais entre os componentes, ilustrando a maior densidade de núcleos de processamento presentes na GPU em comparação à CPU.

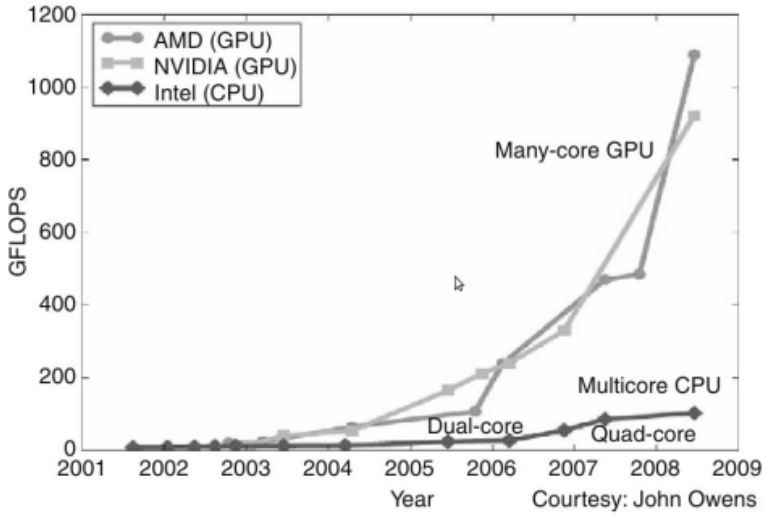

Figura 1.7: Comparação CPU x GPU.
Fonte: (DOMINGOS,)

A GPU permite que aplicativos com alta demanda de processamento sejam execu-

tados em paralelo, graças aos núcleos CUDA. Estes núcleos estão em maior quantidade do que os núcleos de processamento da CPU e, quando combinados com a alta capacidade de paralelismo, atingem performances cada vez mais superiores e eficientes (NICKOLLS; DALLY, 2010).

Neste trabalho, para uma melhor utilização dos núcleos CUDA, o algoritmo foi implementado na linguagem Python. A biblioteca PyTorch foi fundamental para a abstração, tendo em vista sua capacidade de transferir dados e executar instruções em GPU de maneira simplificada, resultando no seguinte fluxo:

- Realiza a alocação de memória em GPU;
- Envia o dado selecionado da CPU ao espaço de memória alocado em GPU;
- Executa determinadas operações otimizadas em GPU;
- Retorna o resultado processado na GPU;
- Envia o dado da GPU de volta à CPU para análise.

1.6 Organização do Trabalho

Este trabalho está organizado da seguinte forma: o Capítulo 2 apresenta a revisão bibliográfica e os conceitos fundamentais. O Capítulo 3 detalha o desenvolvimento da metodologia e a implementação proposta. O Capítulo 4 descreve os parâmetros de teste, bem como os resultados obtidos e suas respectivas discussões. Por fim, o Capítulo 5 apresenta as conclusões e sugestões para trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

2.1 Fundamentação Teórica

2.1.1 Genômica Viral e Variantes Virais

A genômica viral é uma das principais ferramentas no campo da saúde, essencial para o monitoramento de novas doenças, evidenciado principalmente pela pandemia viral do SARS-CoV-2 nos últimos anos (PUJOL, 2023). Para o desenvolvimento de uma vacina e para compreender a evolução dos patógenos, evitando maior perigo à sociedade, este campo torna-se cada vez mais essencial. A capacidade de transcrever o genoma de um vírus, conhecida como Sequenciamento de Biossequências, trata-se de um conjunto de métodos bioquímicos cujo objetivo é determinar a ordem exata em que os nucleotídeos (A, C, G e T, no caso do DNA) aparecem em uma molécula, permitindo que cientistas e autoridades de saúde respondam a surtos com velocidade e precisão. Na Figura 2.1, é possível observar uma representação esquemática desse fluxo, detalhando as etapas fundamentais para a obtenção da informação genética a partir da amostra biológica.

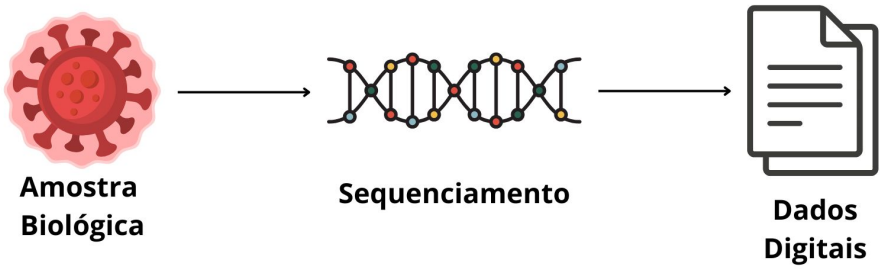

Figura 2.1: Representação das etapas do sequenciamento.
Fonte: Elaborada pelo autor

Essa sequência gerada, também chamada de genoma, é o que codifica e dá ao vírus suas características, que podem ser identificadas como DNA ou RNA. Esse genoma, no entanto, não é estático. Durante o processo de replicação viral, erros podem ocorrer, resultando em mutações. As mais comuns são:

- Mutações de Ponto (SNPs): A alteração de um único nucleotídeo na sequência.
- Inserções e Deleções (Indels): A adição ou remoção de um ou mais nucleotídeos.

Embora muitas dessas alterações sejam neutras, algumas podem modificar fatores fundamentais do vírus, como sua transmissibilidade, virulência ou capacidade de evasão imune (RODRIGUES, 2023). Com a mutação se consolidando, aquele vírus se fixa e gera uma nova população viral, originando, assim, uma nova variante viral. A identificação de tais variantes, realizada através da vigilância genômica, é um dos pontos centrais deste trabalho. Este processo de vigilância, por sua vez, depende da

nossa capacidade de processar um imenso volume de amostras. Mas como a informação contida em um vírus é convertida em dados digitais que podem ser analisados por um computador? A resposta está nas tecnologias de Sequenciamento do Genoma, o alicerce técnico de toda a genômica moderna, cujo desenvolvimento e evolução serão detalhados a seguir.

2.1.2 Sequenciamento Genômico

A tecnologia fundamental que transforma a molécula de um vírus em dados analisáveis é o sequenciamento, um processo que evoluiu drasticamente ao longo das décadas. Ele converte a biossequência (DNA ou RNA) em informação digital, a ordem exata dos nucleotídeos. Uma vez que o genoma viral foi sequenciado, a análise genômica computacional torna-se possível e necessária para extrair significado biológico desses dados e entender a correlação entre genomas virais, evidenciando pontos de mutação para compreender como essa alteração afetou características fundamentais do vírus.

A Primeira Geração: O Sequenciamento de Sanger

O método de sequenciamento por terminação de cadeia, desenvolvido por Frederick Sanger na década de 1970, foi o pilar da genômica por quase trinta anos (SANGER; NICKLEN; COULSON, 1977). Considerado o "padrão-ouro" em termos de precisão, o método Sanger gera uma única leitura de sequência (*read*) por vez, que pode ser relativamente longa, mas compensa pela sua altíssima qualidade. O método funciona da seguinte maneira:

1. **Preparação da Reação:** A fita dupla de DNA que será lida é separada em duas fitas simples. Um *primer* se liga na fita que servirá de molde, indicando o ponto de partida para a cópia.

2. **Cópia e Terminação:** A amostra é dividida em quatro tubos. Em cada tubo, uma enzima começa a copiar a fita de DNA. Além dos "tijolos"de DNA normais, cada tubo contém uma pequena quantidade de um "tijolo terminador"específico (A, T, C ou G). Quando esse tijolo terminador é usado, a cópia para. O resultado é uma coleção de fragmentos de DNA de todos os tamanhos possíveis em cada tubo.

3. **Separação e Leitura:** Os fragmentos dos quatro tubos são colocados em um gel e separados por tamanho usando eletricidade. A sequência de DNA é então lida diretamente no gel, de baixo para cima, observando em qual coluna (A, T, C ou G) a próxima "banda"aparece.

Apesar de sua precisão excelente, o método Sanger possui uma principal limitação: seu baixo rendimento. Como ele processa apenas um fragmento de DNA por reação, seu custo por base é elevado e o processo é lento para projetos de grande escala. Esse gargalo tornou o sequenciamento de genomas completos ou a análise de milhares de amostras, como em uma epidemia, uma tarefa impraticável, o que impulsionou o desenvolvimento de novas tecnologias para superar essas dificuldades, que seriam conhecidas como Sequenciamento de Nova Geração (NGS).

Sequenciamento de Nova Geração (NGS)

Com o objetivo de superar o gargalo de alto custo do método de Sanger, uma nova onda de tecnologias surgiu no início dos anos 2000, também conhecidas como Sequenciamento de Nova Geração (NGS). Em vez de uma revolução em um único método, o NGS representa uma mudança de paradigma, saindo de reações individuais para uma abordagem de paralelismo massivo. O sequenciamento do primeiro genoma humano teve um custo estimado de 3 bilhões de dólares e levou 12 anos; já com o NGS, estima-se um custo em torno de 1000 dólares e a duração de um único dia (MUZZEY; EVANS; LIEBER, 2015).

O princípio fundamental do NGS é a capacidade de sequenciar milhões ou até bilhões de fragmentos de DNA simultaneamente de uma única vez. Essa capacidade de processamento em paralelo diminuiu drasticamente o custo por base sequenciada e aumentou a velocidade de geração de dados em várias ordens de magnitude. Foram os contínuos avanços nessas tecnologias de alto rendimento que impulsionaram o crescimento exponencial dos bancos de dados de sequências biológicas. O NGS não diz respeito a uma única técnica, e sim a uma coleção delas pós-Sanger, cada uma com sua própria tecnologia (MUZZEY; EVANS; LIEBER, 2015), podendo ser baseada em:

- Sequenciamento por Síntese (Illumina): Milhões de fitas de DNA são copiadas ao mesmo tempo. A cada passo, uma peça colorida (A, T, C ou G) é adicionada, e uma câmera tira uma foto para ver a cor de cada fita.
- Sequenciamento por Ligação (SOLiD): Em vez de adicionar uma peça de cada vez, a máquina testa pequenas sondas coloridas e vê qual delas se encaixa. A cor da sonda revela um par de bases da sequência.
- Sequenciamento por Semicondutor de Íons (Ion Torrent): O DNA fica em milhões de poços minúsculos, cada um com um sensor de pH (acidez). Quando uma peça é adicionada, ela libera um íon que muda o pH, e o sensor detecta essa mudança.

Na Figura 2.2, é possível observar o contraste entre essas abordagens, evidenciando como a paralelização do NGS otimiza as etapas em comparação ao método linear de Sanger.

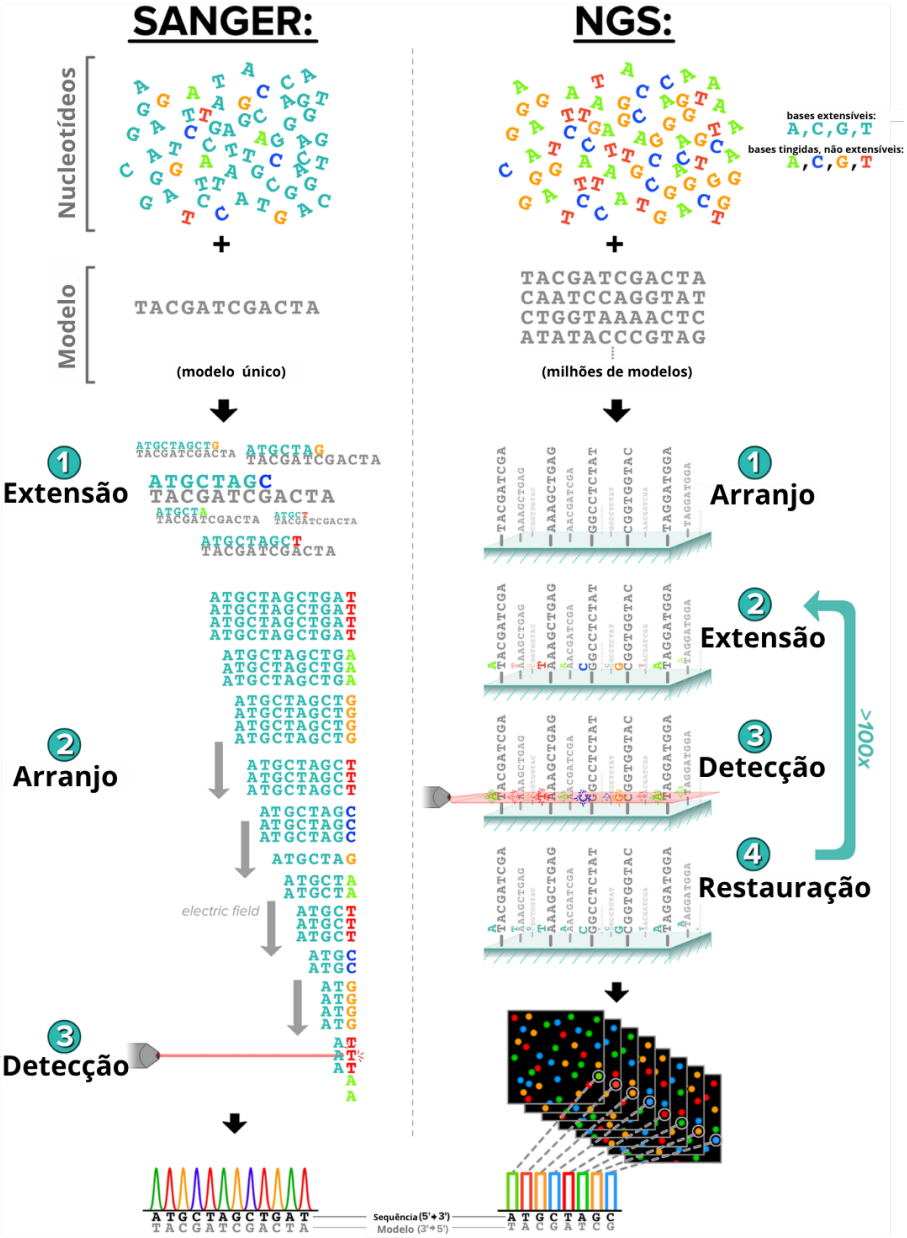

Figura 2.2: Representação das etapas do método de Sanger em comparação ao Sequenciamento de Nova Geração.
Fonte: Adaptado de (MUZZEY; EVANS; LIEBER, 2015)

Independente de qual tecnologia de sequenciamento seja utilizada, o resultado será o mesmo, uma quantidade massiva de leituras curtas (*short reads*) de uma biossequência. Cada *short read* representa um pequeno fragmento do determinado genoma desconectado de um grande quebra-cabeça, tornando necessária a remontagem desse genoma antes de realmente analisar e decidir algo. O principal e fundamental método para essa remontagem é determinar a posição exata daquele fragmento dentro do grande conjunto. Esse processo de mapeamento da posição é chamado de alinhamento de sequências, que envolve a computação e comparação de milhares dessas várias *short reads* para assim gerar o resultado final. Porém, essa atividade é de alto custo computacional, portanto, a otimização dos algoritmos para a implementação dessa atividade é fundamental.

2.1.3 Alinhamento de Sequências

O alinhamento de sequências é uma das atividades fundamentais da bioinformática. O que ele busca fazer é a comparação de duas biossequências para entender igualdades e divergências entre ambas, perfeito para identificar possíveis variantes virais resultantes de alguma mutação. O alinhamento fundamentalmente é tratado de duas formas:

Alinhamento Local

O alinhamento local consiste na separação das sequências em fragmentos de tamanho predeterminado para o problema. Com os vários pedaços do genoma separados, eles são comparados conforme a sua posição em relação à sequência. Essa abordagem é extremamente poderosa do ponto de vista de identificação de regiões conservadas da sequência, compreendendo como aquela característica se manteve conforme a mutação.

O principal algoritmo que realiza essa comparação é o de Smith-Waterman

(SMITH; WATERMAN, 1981). Ele utiliza programação dinâmica para identificar essas similaridades de maneira otimizada, por meio da construção de uma tabela de pontuações baseada na comparação entre as bases nitrogenadas. Dentro das opções de comparação, temos:

- ***Match***: Comparação entre duas bases iguais (identidade).
- ***Mismatch***: Comparação entre duas bases diferentes (substituição).
- ***Gap***: Comparação entre uma base e uma lacuna, representada pelo caractere '-'. Representa uma inserção ou deleção (*indel*) na sequência evolutiva.

Na Figura 2.3, é possível visualizar exemplos práticos dessas três situações no contexto de um alinhamento.

Figura 2.3: Exemplos de casos de *Match*, *Mismatch* e *Gap*.
Fonte: (GOMES, 2021)

Para cada uma das comparações, determinada pontuação é atribuída, normalmente com pontuação positiva para um *match* e negativa para *mismatch* e *gap*. Após a definição das pontuações, temos a construção da tabela:

- Considerando uma comparação entre duas sequências distintas de tamanho total M e N.

- A primeira linha e coluna da matriz serão inicializadas com valor 0 como valor padrão.
- Para cada uma das posições, a serem calculadas, será feita uma seleção entre o maior valor das 3 seguintes opções:
  - Valor da posição diagonal superior à esquerda + Pontuação do *Match*/*Mismatch*.
  - Valor da posição de cima + Penalidade por *gap*.
  - Valor da posição à esquerda + Penalidade por *gap*.
- Portanto, para cada posição de uma matriz H, na linha i e coluna j, temos o seguinte valor:

$$H(i,j) = \max \begin{cases} H(i-1,j-1) + \sigma(q_{i-1},s_{j-1}) & (\textit{Match/Mismatch}) \\ H(i-1,j) + \text{gap} & (\text{Deleção}) \\ H(i,j-1) + \text{gap} & (\text{Inserção}) \\ 0 & (\text{Zero}) \end{cases}$$

Com a matriz preenchida, o maior valor global é identificado. A partir deste ponto, inicia-se o processo de *traceback*, retrocedendo em direção às células antecessoras (diagonal, vertical ou horizontal) que originaram a pontuação atual, até encontrar uma célula com valor zero. A Figura 2.4 exemplifica uma matriz de pontuação calculada, onde os valores numéricos determinam esse caminho de volta. Ao concluir o processo, o alinhamento local ótimo é reconstruído e apresentado, sendo possível, posteriormente, recalcular a matriz para encontrar outros alinhamentos locais secundários.

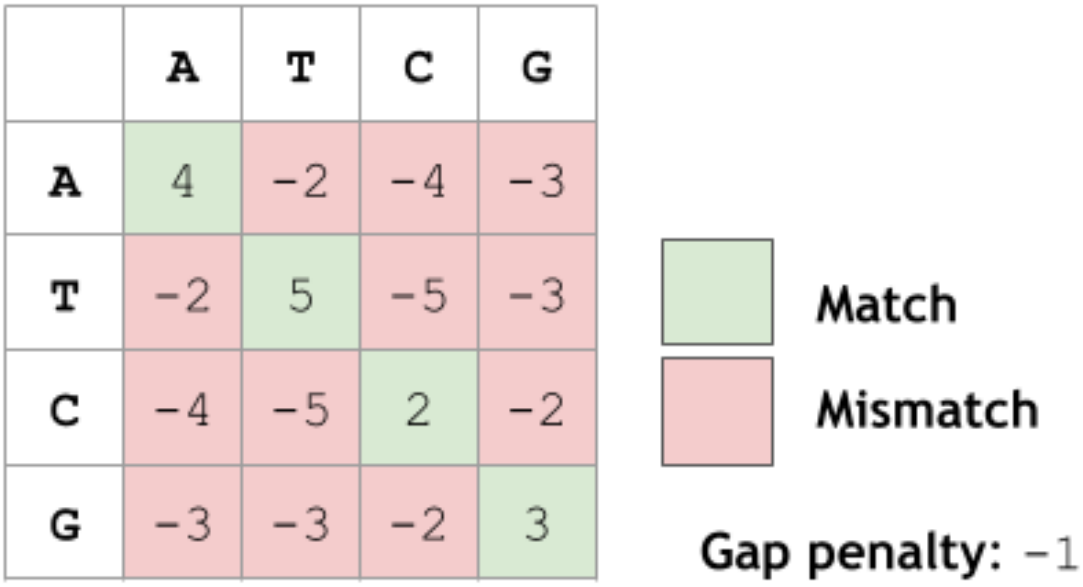

Figura 2.4: Exemplo de Tabela de Pontuações.
Fonte: (GOMES, 2025)

Alinhamento Global

O alinhamento global, em contrapartida, consiste na análise de toda a sequência. Ele busca ao longo de toda a extensão de ambas as sequências comparar e encontrar o maior valor de similaridade entre elas. Para essa comparação, a inserção de *gaps* ao longo das sequências é fundamental, pois permite que o máximo de *matches* seja reproduzido, proporcionando a melhor pontuação. Na Figura 2.5, é possível visualizar a diferença de abrangência entre essa abordagem e o alinhamento local abordado anteriormente.

O algoritmo que faz esse tipo de alinhamento é o Needleman-Wunsch (NEEDLEMAN; WUNSCH, 1970). Ele também utiliza programação dinâmica e calcula o melhor alinhamento conforme a tabela de pontuação. A construção da tabela é feita da mesma maneira que no algoritmo anterior; a principal diferença reside na maneira pela qual o *traceback* é executado.

Nesse algoritmo, como um alinhamento global é necessário, sem a possibilidade

de interrupção no momento em que uma pontuação 0 é atingida, o algoritmo precisa iniciar sempre da última posição da matriz e seguir até o topo, mesmo acumulando uma pontuação negativa. O processo de *traceback* então é diferenciado da seguinte maneira:

- Caso o movimento seja para a diagonal, em $H(i-1, j-1)$, ocorre uma situação de *match*/*mismatch*. As posições são mantidas e $H(i-1, j-1)$ vira a nova posição de comparação.
- Caso o movimento seja para cima, em $H(i-1, j)$, ocorreu uma situação de *gap* na sequência vertical. A base da sequência horizontal é mantida e $H(i-1, j)$ vira a nova posição de comparação.
- Caso o movimento seja para a esquerda, em $H(i, j-1)$, ocorreu uma situação de *gap* na sequência horizontal. A base da sequência vertical é mantida e $H(i, j-1)$ vira a nova posição de comparação.

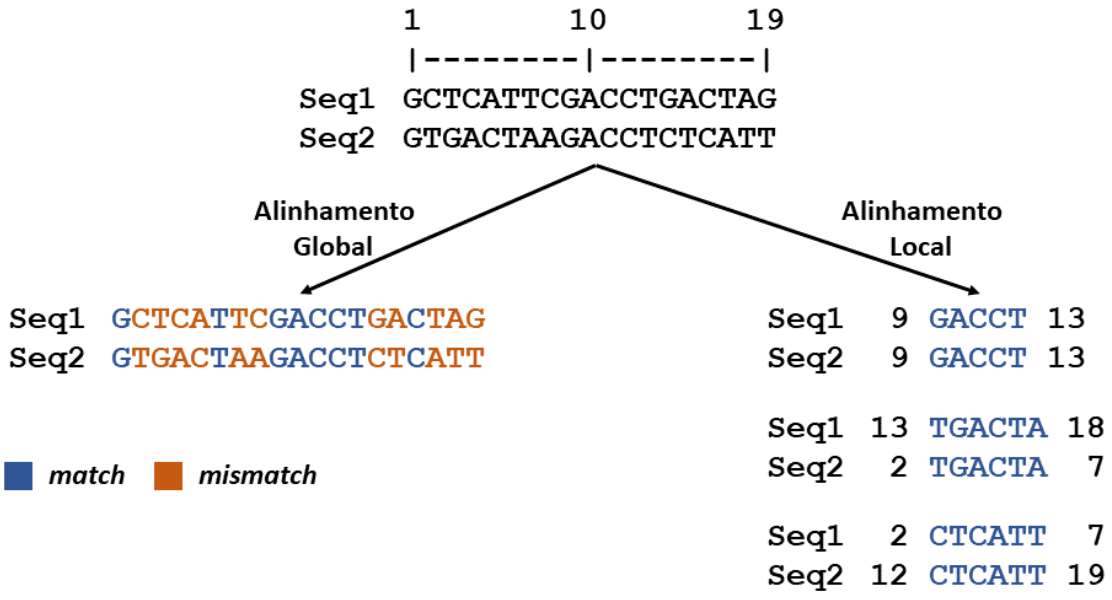

Figura 2.5: Comparação entre Alinhamento Global e Local.
Fonte: (AMORIM, 2022)

Os algoritmos de programação dinâmica, como Needleman-Wunsch e Smith-Waterman, garantem a obtenção do alinhamento ótimo, seja ele global ou local. Contudo, sua aplicação se restringe à comparação de apenas duas sequências por vez. Em cenários de análise genômica real, onde frequentemente é necessário comparar múltiplas sequências simultaneamente — por exemplo, para estudar uma família de genes ou a evolução de um vírus —, essa abordagem par-a-par se torna insuficiente. Para suprir essa necessidade, foi introduzido o campo do Alinhamento Múltiplo de Sequências (AMS).

2.1.4 Alinhamento Múltiplo de Sequências

O Alinhamento Múltiplo de Sequências (AMS) segue o mesmo princípio dos alinhamentos de pares que foram apresentados, porém agora para uma quantidade maior, envolvendo de três a milhares de sequências simultaneamente. A Figura 2.6 ilustra esse conceito, onde é possível visualizar a conservação de nucleotídeos ao longo de diversas amostras. Diferente dos algoritmos determinísticos, o AMS é associado a abordagens heurísticas ou probabilísticas, buscando a maior significância biológica possível para todas as sequências dispostas. As abordagens que se destacam para este caso são o alinhamento progressivo e o iterativo.

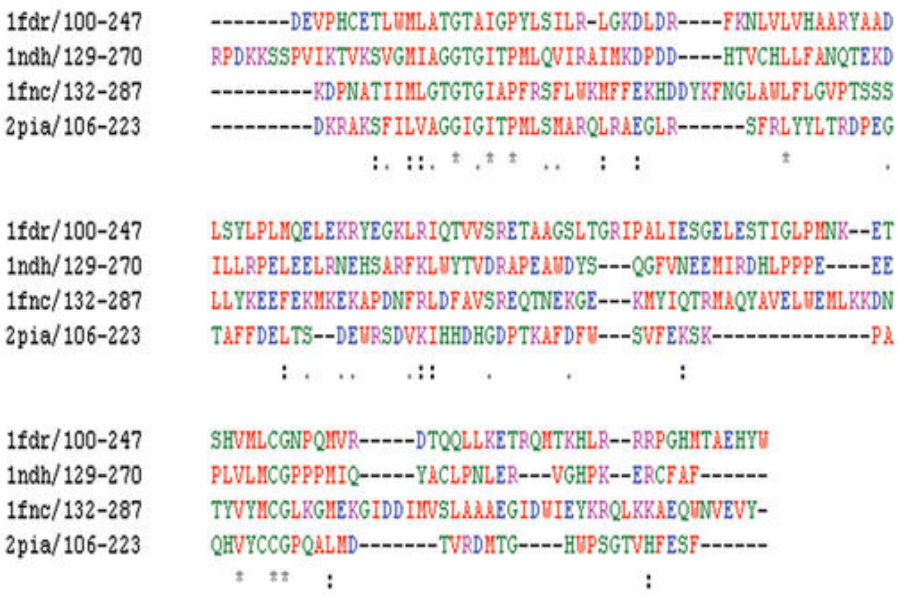

Figura 2.6: Exemplo de Alinhamento Múltiplo de Sequências.
Fonte: (CHENNA et al., 2003)

Alinhamento Progressivo

O alinhamento progressivo é uma estratégia heurística que propõe adicionar sequências conforme o alinhamento avança. A formulação origina-se de Hogeweg e Hesper (HOGEWEG; HESPER, 1984), na qual foi proposto um modelo em árvore para ser usado como base. Nessa árvore, as folhas são as sequências a serem alinhadas, e os nós internos são alinhamentos parciais entre as sequências realizados durante a execução. Ao final de todas essas execuções, temos na raiz o resultado de todos os alinhamentos.

Essa abordagem é a base para várias ferramentas amplamente utilizadas dentro do campo de alinhamentos, tais como Kalign (DEOROWICZ; DEBUDAJ-GRABYSZ; GUDYś, 2014), T-COFFEE (NOTREDAME; HIGGINS; HERINGA, 2000) e a família Clustal, destacando o ClustalW (THOMPSON; HIGGINS; GIBSON, 1994). Uma das principais vantagens desse método é a qualidade dos resultados sem um alto custo computacional. Porém, devido à sua natureza gulosa, erros em alinhamentos iniciais acabam impactando o resto da execução, afetando negativamente execuções posterio-

res.

Alinhamento Iterativo

Já o alinhamento iterativo é uma estratégia que parte de um alinhamento já realizado, e cada iteração tenta aperfeiçoá-lo. Alguns critérios podem ser utilizados para definir o ponto de parada do alinhamento, tais como a quantidade de iterações ou alguma característica específica esperada.

Esse método tem como principal vantagem a correção de erros causados em alinhamentos anteriores, reavaliando, modificando e, consequentemente, melhorando sua qualidade na maioria dos casos. Alguns exemplos de algoritmos que fazem uso desse princípio são os algoritmos bioinspirados, como o algoritmo genético (NOTREDAME; HIGGINS, 1996), colônia de abelhas (RUBIO-LARGO; VEGA-RODRÍGUEZ; GONZÁLEZ-ÁLVAREZ, 2016) e colônia de formigas (LEE et al., 2008).

Contudo, este método também possui desvantagens. Pode-se chegar a um ponto máximo local de otimização, gerando uma estagnação e impossibilidade de convergência para a resposta ótima. Outra principal desvantagem é a questão de que, quanto maior a qualidade desejada, mais iterações precisam ser realizadas; logo, mais recursos computacionais são necessários, impactando diretamente o desempenho do algoritmo.

Para superar este gargalo de desempenho e possibilitar a análise dessa grande quantidade de dados, a aplicação de computação paralela tornou-se essencial. Dentre elas, as Unidades de Processamento Gráfico (GPUs) são as que mais vêm se destacando como solução de alto impacto e custo reduzido se comparado às CPUs na questão de desempenho, graças à sua arquitetura de núcleos CUDA e sua alta capacidade de paralelização de processamento.

2.1.5 Arquitetura de GPUs e CUDA

Para superar os gargalos dentro do campo da bioinformática, a Computação de Propósito Geral em Unidades de Processamento Gráfico (GPGPU) surgiu como uma solução poderosa aos processamentos realizados totalmente em CPUs. Como alternativa à execução sequencial e em baixa latência dos processadores, as GPUs foram projetadas com uma arquitetura que possibilita o paralelismo massivo de atividades, viabilizando uma grande vazão de dados e execução de operações independentes de maneira simultânea. A Figura 2.7 ilustra essa diferença estrutural, contrapondo a área dedicada a controle e cache na CPU versus a densidade de unidades de cálculo na GPU.

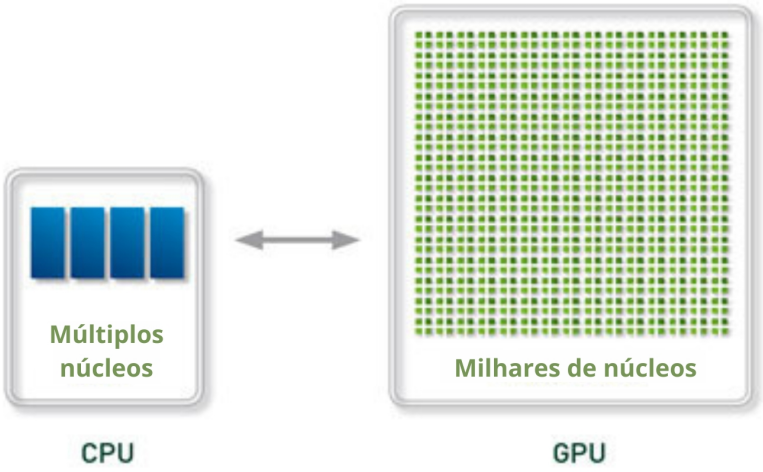

Figura 2.7: Comparação da arquitetura de CPU x GPU.
Fonte: Adaptado de (GHORPADE et al., 2012)

Pensando nesse aproveitamento das GPUs além do poder gráfico, a arquitetura CUDA (Compute Unified Device Architecture) utilizada nas placas de vídeo NVIDIA permite que os desenvolvedores acessem essa execução simultânea. As rotinas de atividades a serem executadas são escritas em formato de *kernels*, e esses *kernels* são executados pelos chamados *grids*, que são organizações de blocos de *threads*, com a

possibilidade da comunicação e cooperação entre esses blocos, mas sendo cada um deles executados dentro da unidade de hardware da GPU chamada *Streaming Multiprocessor* (SM). Dentro de cada bloco, temos várias organizações de 32 *threads*, e cada uma dessas é chamada de *warp*, podendo variar sua quantidade conforme a quantidade máxima de *threads* suportada por bloco pela GPU.

As *warps* são fundamentais para tornar a GPU uma arquitetura essencial na execução, pois cada uma delas executa certa instrução, no modelo também chamado SIMT (*Single Instruction, Multiple Threads*), permitindo para cada unidade apenas a realização de cálculo, sem a necessidade de manter controle sobre as instruções, tornando a execução muito mais performática e de escala massiva. Em conjunto com essa execução de várias *threads* para uma única instrução, temos o uso de dois tipos principais de memória para compartilhamento de dados: a global, que é maior e mais lenta; e a compartilhada, que é menor, mas mais rápida.

Porém, a máxima performance a ser atingida com a execução em GPU não depende unicamente de como a arquitetura funciona como um todo, nem o quão potente o *hardware* é, mas sim da adaptabilidade do algoritmo ao paralelismo massivo, pois caso um algoritmo sem qualquer adaptação à execução paralela em CUDA seja executado, o resultado deve permanecer o mesmo se comparado à CPU. Um exemplo para este problema são as redes *Bipartite*, que possuem acesso irregular à memória, levando a baixo desempenho, principalmente em GPU. Para solucionar isso, uma abordagem simples e eficaz é a aplicação de redes *One-Step*, com acesso mais simples e direto, tornando a execução muito mais performática e acessível para execução paralela.

2.1.6 Análise de Redes: *Bipartite* vs. *One-Step*

Após todo o *pipeline* computacional para identificação de variantes virais e otimização da execução com paralelismo, a última etapa é a análise desses dados para identificação das relações entre as sequências. Para esta análise, a Teoria dos Grafos

é uma disciplina fundamental, pois esta fornece ferramentas poderosas para análise visual e analítica por meio da construção de redes de interação.

Uma das maneiras de realizar essa análise é por meio de uma rede *Bipartite* (também chamada de dois modos). Nessa rede, dois tipos de nós são utilizados para análise: amostras de vírus e as mutações identificadas. Depois dessa categorização dos nós, as ligações, representadas por arestas, são realizadas entre os dois nós diferentes (vírus e mutações), mostrando qual é a origem daquela determinada mutação.

Mas, apesar de realizar uma representação completa de toda a rede, a visualização é complexa, tendo em vista a grande quantidade de nós e arestas sendo representados em uma rede, principalmente dentro do campo da bioinformática com as milhares de sequências sendo alinhadas. Pensando em uma solução para isso, surgem as redes *One-Step*, principal foco de otimização dentro deste trabalho. Essa rede possui como objetivo principal a criação de uma nova rede, que foca principalmente na relação de similaridade entre os vírus.

Neste novo tipo de rede, aplicamos apenas um tipo de nó, sendo este as amostras virais. A representação por meio das arestas é mantida, porém, com uma condição adicionada: a ligação só ocorre caso uma mutação em comum seja compartilhada entre as amostras. Com essa nova análise, uma visualização mais clara e direta pode ser realizada, tornando a análise muito mais concisa e facilitada. Com este último passo de construção da rede, a última etapa de visualização dos dados é concluída, permitindo a utilização deste mapa visual para uma análise direta e simplificada das variantes virais.

2.2 Trabalhos Correlatos e Estado da Arte

Em arquiteturas de CPU, o estado da arte na otimização do algoritmo de Smith-Waterman é exemplificado pela ferramenta SWIMM2.0 (RUCCI et al., 2019). A abordagem maximiza a performance em processadores Intel ao explorar as extensões veto-

riais avançadas AVX-512, permitindo processamento de múltiplos dados em um único ciclo de *clock* (SIMD), tornando-o um dos alinhadores baseados em CPU mais rápidos. No contexto de aceleração por hardware, ferramentas como esta servem como um importante ponto de referência para quantificar os ganhos de desempenho obtidos pelas implementações em GPU.

Para superar as limitações das CPUs, as pesquisas na área de desempenho em alinhamento de sequências avançaram para arquiteturas massivamente paralelas, tendo no CUDASW++ 4.0 (SCHMIDT et al., 2024) sua principal referência. Esta ferramenta estabelece o padrão de desempenho para o algoritmo de Smith-Waterman ao explorar instruções intrínsecas de hardware e otimizar a comunicação intra-*warp*. O trabalho evidencia que, em ambientes de GPU, o ganho de performance não provém apenas do número elevado de núcleos, mas do refinamento no acesso à memória e no uso eficiente de instruções SIMT (*Single Instruction, Multiple Threads*). A Figura 2.8 ilustra essa disparidade, apresentando o CUDASW++ em destaque em comparação a outras soluções.

Outro caso é o ADEPT (AWAN et al., 2020), que já demonstrava o potencial das GPUs para o alinhamento de sequências. A ferramenta propôs uma estratégia de alinhamento que funciona tanto para sequências de DNA quanto para proteínas, fazendo uso da GPU para otimização do tempo de execução. Porém, ferramentas desse tipo frequentemente não conseguiam extrair todo o potencial das GPUs modernas, sendo o acesso à memória um problema recorrente e representando uma grande fatia desse gargalo de desempenho. Ao mesmo tempo, essa abordagem demonstra a importância do desenvolvimento de algoritmos otimizados para fazer o melhor uso das arquiteturas mais recentes no mercado.

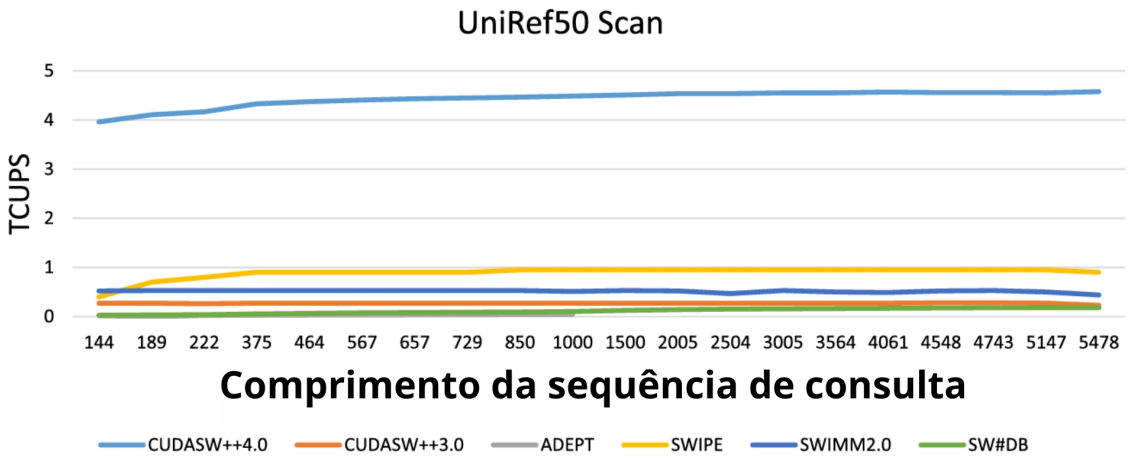

Figura 2.8: Comparação entre vários algoritmos de alinhamento em CPU e GPU.
Fonte: Adaptado de (SCHMIDT et al., 2024)

Capítulo 3

Desenvolvimento e Plano de Trabalho

Com os grandes avanços na biologia e o crescimento da quantidade de dados genômicos, a metodologia utilizada no desenvolvimento deste projeto tem como fundamentação um levantamento bibliográfico baseado no tema de aplicação de métodos e estratégias computacionais para a análise de variantes virais, de modo a conseguir um melhor desempenho e aplicar o que há de mais recente em tecnologia.

3.0.1 Redes *One-Step*

Para a realização da análise das variantes de maneira mais detalhada e concisa, foram utilizadas Redes *One-Step*. No cálculo da rede, houve uma conexão de duas variantes com apenas a diferença de uma base da biossequência, tornando a visualização mais otimizada para uma análise profunda. Além desta visualização otimizada, tem-se um total de $n(n-1)/2$ pares possíveis.

Para a análise de várias biossequências e suas diferenças de bases, a GPU foi utilizada, uma vez que esta técnica facilita a aplicação de paralelismo, ideal para a construção das Redes *One-Step* e as comparações entre as diferentes sequências.

Após esta comparação, uma série de dados disponibilizados pelo *Centers for Disease Control* (CDC) foram introduzidos para uma análise minuciosa. Foram executados

testes com quantidades variadas de sequências, sendo os tempos de execução medidos e armazenados. Ao final, após uma análise dos tempos medidos previamente, uma avaliação de significância biológica foi aplicada, buscando compreender qual a variação aplicada e como isso afeta o vírus.

Após encontrar o alinhamento ótimo, tem-se, em teoria, a melhor biossequência formada pelos diversos alinhamentos diferentes. Com isso, agora é necessário encontrar a diferença entre a sequência original e sua variante, mas sem conhecimento da sequência que originou esta variante gerada pelo algoritmo. Para isso, então, utiliza-se as redes *One-Step*. O método *One-Step* consiste em conectar sequências estreitamente relacionadas sem qualquer suposição sobre sua origem ancestral (ANDRADE, 2020), apresentando visualmente uma relação de hereditariedade entre as várias sequências analisadas.

A construção de redes *One-Step* é fundamental para a análise das variantes virais, tendo em vista que esta realiza a conexão entre as diferentes biossequências, possibilitando identificar pontos de alteração e com propensão à mutação, apontando a origem daquela nova característica da variante (CAETANO et al., 2016).

3.0.2 Unidades de Processamento Gráfico (GPU)

Para uma execução mais eficiente das análises que foram realizadas durante o projeto, tal qual a construção das redes *One-Step*, demanda-se grande poder de processamento devido à grande quantidade de sequências a serem analisadas. Por muito tempo, esse processo foi exclusivamente realizado pela Unidade Central de Processamento (CPU) por ser a única unidade para este processamento dentro das máquinas. Porém, com o advento das Unidades de Processamento Gráfico (GPUs), tornou-se possível a execução de várias tarefas simultaneamente, processo também conhecido como paralelismo, com maior eficiência (DOMINGOS,). Mesmo com a introdução de multiprocessadores (processadores com múltiplos núcleos), o desempenho das

GPUs provou-se superior.

A GPU permite que aplicativos com alta demanda de processamento sejam executados em paralelo, graças aos núcleos CUDA. Estes núcleos estão em maior quantidade do que os núcleos de processamento da CPU e, quando combinados com a alta capacidade de paralelismo, atingem performances cada vez mais superiores e eficientes (NICKOLLS; DALLY, 2010).

3.0.3 Implementação do método

Neste projeto, para uma melhor utilização dos núcleos CUDA, a linguagem de programação Python foi escolhida para a implementação. Foi utilizada a biblioteca PyTorch para a melhor abstração e configuração dos testes, tendo em vista sua facilidade na troca entre os ambientes CPU e GPU. A biblioteca permite uma execução simplificada de CUDA, enviando dados para os chamados tensores, nos quais os cálculos e comparações são realizados diretamente pelos núcleos CUDA de maneira paralela e eficiente. A Figura 3.1 ilustra a arquitetura do sistema desenvolvido, demonstrando como essas tecnologias se integram para processar os dados.

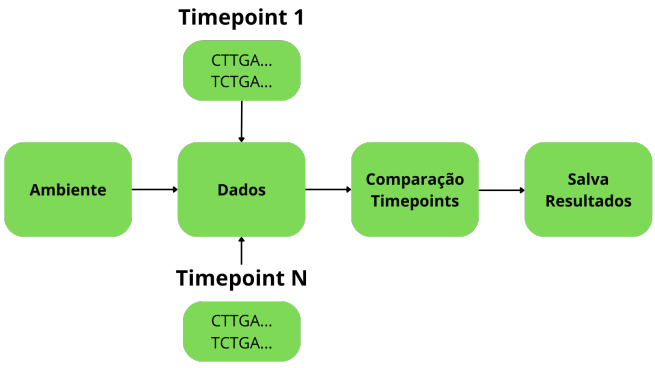

Figura 3.1: Arquitetura do sistema.
Fonte: Elaborada pelo autor

Capítulo 4

Testes e Resultados

4.1 Realização dos testes

Os testes foram realizados utilizando um *dataset* fornecido pelo *Center for Disease Control* (CDC), com um total de 2446 biossequências e tamanho médio de 264 caracteres por sequência. A execução foi realizada em 2 ambientes distintos:

Ambiente 1

- CPU: 12th Gen Intel(R) Core(TM) i5-12400F (2.50 GHz)
- GPU: NVIDIA GeForce GTX 1060 6GB
- Memória: 16 GB
- Sistema Operacional: Windows 11 com WSL 2 - Ubuntu-22.04
- Software: Python 3.10.12, PyTorch 1.13.1+cu117

Ambiente 2

- CPU: 13th Gen Intel(R) Core(TM) i7-13700 (2.10 GHz)
- GPU: NVIDIA GeForce RTX 4060 8GB

- Memória: 32 GB
- Sistema Operacional: Windows 11 com WSL 2 - Ubuntu-22.04
- Software: Python 3.10.12, PyTorch 2.4.0+cu121

Para cada um dos quatro cenários, um *script* de análise foi executado 20 vezes consecutivas, pensando em garantir a confiabilidade estatística dos resultados e permitir a análise da variabilidade do desempenho. A métrica coletada foi o tempo de execução em segundos, medido por meio da função `time.perf_counter()` da biblioteca padrão do Python. Esta abordagem garante que apenas o tempo de processamento do algoritmo seja medido, excluindo os tempos de inicialização e de carregamento dos dados em memória. A média, o desvio padrão, o SPM (Sequências Por Milissegundo) e o *speedup* obtidos serão apresentados e discutidos na seção seguinte.

Por fim, para a representação gráfica dos dados, a biblioteca Matplotlib foi utilizada, demonstrando os tempos aproximados das vinte execuções e uma comparação direta entre os quatro ambientes, mostrando os diferentes tempos de execução.

4.2 Resultados

Os dados quantitativos obtidos, detalhados na Tabela 4.1 e visualizados na Figura 4.1, evidenciam a superioridade do processamento paralelo frente à abordagem sequencial tradicional.

Ambiente	**Plataforma**	**Tempo Médio (s)**	**Desvio Padrão (s)**	**SPM**	**Speedup**
Ambiente 1	CPU	1.8277	0.0650	1.34	-
	GPU	1.2182	0.0290	2.01	1.50x
Ambiente 2	CPU	1.4614	0.0769	1.67	-
	GPU	0.8603	0.0185	2.84	1.70x

Tabela 4.1: Resultados comparativos de desempenho entre CPU e GPU. A tabela mostra o tempo médio de execução, desvio padrão, a quantidade de sequências processadas por milissegundo (SPM) e o *speedup* (aceleração) da GPU.

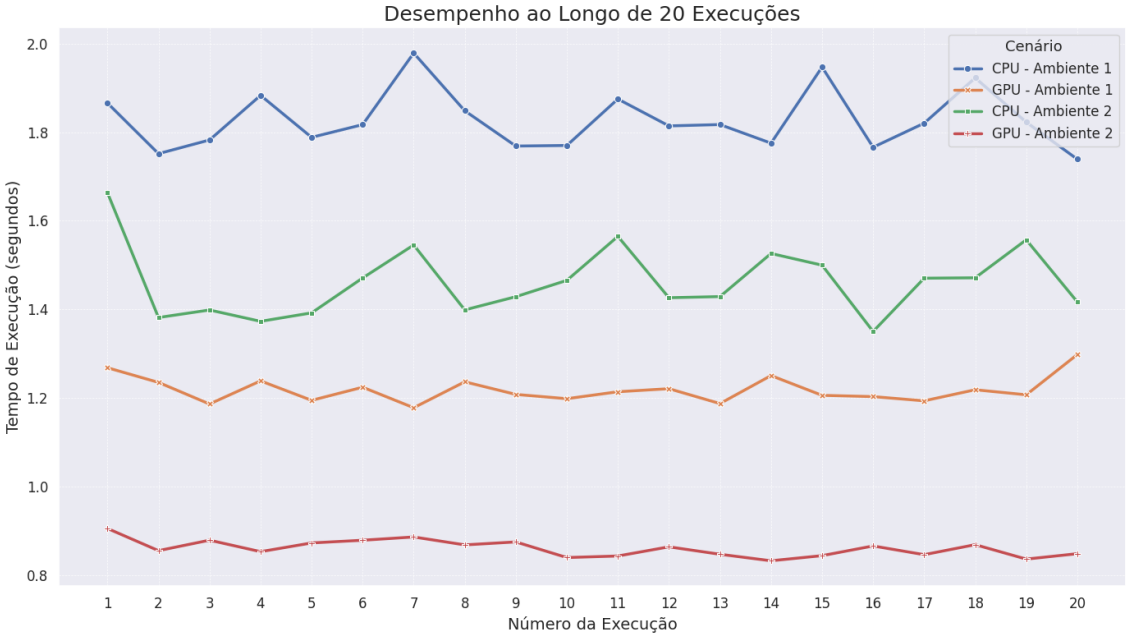

Figura 4.1: Resultados 20 Execuções CPU x GPU.
Fonte: Elaborada pelo autor

Observa-se, primeiramente, uma média de tempo inferior em ambos os cenários para GPU. Este resultado revela um dado contundente sobre a evolução do hardware: a GPU do ambiente 1, a GTX 1060 lançada em 2016, com sua arquitetura Pascal, conseguiu um desempenho superior à CPU do ambiente 2, i7-13700 lançado em 2023, com 16 núcleos e 24 *threads*, demonstrando que mesmo com grande avanço no campo de execução sequencial em CPU, a GPU ainda demonstra desempenho superior em cenários de alta carga de paralelismo.

Também é evidente o ganho de performance obtido se comparado à CPU e GPU dentro dos ambientes. O aumento de 50% e 70% na velocidade, interpretado pelo *speedup* de 1.50x e 1.70x, evidencia muito bem esse cenário. O SPM (Sequências Por Milissegundo) acentua ainda mais essa vantagem, mostrando que o sistema é altamente escalável: em cenários com *datasets* massivos, a solução em GPU conseguiria processar um volume muito maior de dados no mesmo intervalo de tempo.

Outro ponto a ser destacado é a inconsistência do primeiro tempo de todos os cenários. Isso pode ser justificado pela situação de *warm-up* encarada tanto pela CPU

quanto pela GPU, conforme citado em (JR; SKADRON, 2005). Essa situação funciona da seguinte maneira em CPU: a primeira execução busca os dados do *dataset* da memória RAM para as caches do processador (L1/L2/L3), e como o acesso à RAM é relativamente lento se comparado à cache, isso justifica essa lentidão inicial. Em paralelo a esse carregamento, temos outras execuções acontecendo simultaneamente dentro do Sistema Operacional, impactando também na velocidade de execução.

Já em GPU, o cenário envolve outros fatores específicos da biblioteca. Além da transferência de memória da RAM para a VRAM, o PyTorch utiliza uma estratégia de "inicialização preguiçosa" (*lazy initialization*). Na primeira execução, o *framework* precisa criar o contexto CUDA, carregar os drivers e realizar a compilação *Just-In-Time* (JIT) dos *kernels* computacionais. Essa etapa de preparação deixa a primeira rodada mais lenta, mas garante que as instruções subsequentes estejam otimizadas especificamente para aquela GPU, resultando na estabilidade observada nas execuções seguintes.

Em suma, os resultados quantitativos e qualitativos demonstram um ganho de performance significativo com o uso da aceleração por GPU, validando a abordagem e alcançando um *speedup* de até 1.70x, consolidando-a como uma ferramenta eficaz para a interpretação e identificação de agrupamentos de variantes virais. Em conjunto, a aceleração do tempo de resposta e a correta formação dos agrupamentos na rede confirmam a eficácia deste fluxo de análise para a bioinformática.

Capítulo 5

Conclusão

O alinhamento de sequências é uma tarefa fundamental no campo da bioinformática, e a crescente necessidade de analisar grandes volumes de dados genômicos em tempo hábil exige soluções computacionais de alto desempenho. Este trabalho cumpriu seu objetivo principal ao desenvolver e validar uma solução computacional de alto desempenho para o cálculo massivo da distância de Hamming e a construção de Redes *One-Step*, utilizando a arquitetura paralela de GPUs para superar as limitações de tempo de execução das CPUs.

Os resultados obtidos demonstram a hipótese de que a utilização da arquitetura massivamente paralela das GPUs pode reduzir significativamente o tempo de processamento, se comparado a ambientes de execução sequencial. A implementação em GPU alcançou uma aceleração (*speedup*) de até 1.70x em comparação com a execução equivalente em CPU, e atingiu uma taxa de processamento de aproximadamente 2 sequências por milissegundo, demonstrando a viabilidade e o ganho de performance da abordagem. Adicionalmente, a aplicação da metodologia de Redes *One-Step* para a visualização dos dados de variantes se mostrou eficaz na identificação de agrupamentos (*clusters*) que correspondem a linhagens virais conhecidas, confirmando seu valor como ferramenta de análise qualitativa. Em conjunto, os resultados quantitativos e qualitativos demonstram que a estratégia proposta foi bem-sucedida em criar uma

análise mais rápida sem a necessidade de sacrificar a capacidade de extrair conhecimento biológico relevante.

Como trabalhos futuros, propõe-se a expansão desta pesquisa em algumas frentes. Primeiramente, a otimização pode ser estendida para outras etapas do *pipeline* de bioinformática, como a própria chamada de variantes, buscando uma aceleração ainda maior do fluxo completo. Sugere-se também a avaliação da implementação em uma gama mais ampla de arquiteturas de GPU e a comparação com um conjunto maior de *datasets* virais para testar a escalabilidade e generalização do método. Por fim, uma análise em sequências ultralongas (com até milhões de caracteres por sequências) vem ganhando destaque, sendo uma forte possibilidade de otimização para o futuro.

Referências Bibliográficas

AMORIM, A. R. *Abordagens baseadas em consistência para alinhamento múltiplo de sequências de DNA*. Tese (Tese (Doutorado em Ciências da Computação e Matemática Computacional)) — Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, São Carlos, 2022. Disponível em: <https://www.teses.usp.br/teses/disponiveis/3/3141/tde-21032022-142918/publico/AndersonRiciAmorimCorr22.pdf>.

ANDRADE, M. C. Construção de redes one-step para estudos evolucionários utilizando gpu. Universidade Estadual Paulista (Unesp), 2020.

ARAF, Y. et al. Omicron variant of sars-cov-2: genomics, transmissibility, and responses to current covid-19 vaccines. *Journal of medical virology*, Wiley Online Library, v. 94, n. 5, p. 1825–1832, 2022.

ARAÚJO, N. D. de et al. A era da bioinformática: seu potencial e suas implicações para as ciências da saúde. *Estudos de biologia*, v. 30, n. 70/72, 2008.

AWAN, M. G. et al. Adept: a domain independent sequence alignment strategy for GPU architectures. *BMC Bioinformatics*, v. 21, n. 1, p. 1–29, 2020.

CAETANO, D. G. et al. *Caracterização de variantes virais de HIV-1 em indivíduos soropositivos com perfil de controle da progressão para a AIDS e da replicação viralavaliação da ocorrência de variantes de escape da resposta imune*. Tese (Doutorado), 2016.

CARVALHO, M. C. d. C. G. d.; SILVA, D. C. G. d. Sequenciamento de dna de nova geração e suas aplicações na genômica de plantas. *Ciência Rural*, SciELO Brasil, v. 40, p. 735–744, 2010.

CHENNA, R. et al. Multiple sequence alignment with the clustal series of programs. *Nucleic acids research*, Oxford University Press, v. 31, n. 13, p. 3497–3500, 2003.

DEOROWICZ, S.; DEBUDAJ-GRABYSZ, A.; GUDYś, A. Kalign-LCS-A more accurate and faster variant of kalign2 algorithm for the multiple sequence alignment problem. In: *Man-Machine Interactions 3*. [S.l.]: Springer, 2014. p. 495–502.

DOMINGOS, D. P. Utilização de gpu para sistemas de paralelismo massivo.

FIGUEIRÔA, L. H. A. et al. *Uma plataforma híbrida baseada em FPGA para a aceleração de um algoritmo de alinhamento de sequências biológicas*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2015.

GARDY, J. L.; LOMAN, N. J. Towards a genomics-informed, real-time, global pathogen surveillance system. *Nature Reviews Genetics*, v. 19, n. 1, p. 9–20, Jan 2018. Disponível em: <https://doi.org/10.1038/nrg.2017.88>.

GHORPADE, J. et al. Gpgpu processing in cuda architecture. *arXiv preprint arXiv:1202.4347*, 2012.

GOMES, V. Z. *Algoritmo Genético Híbrido Com Refinamento Local Para Alinhamento Múltiplo De Sequências*. 47 p. Dissertação (Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação)) — Universidade Estadual Paulista "Júlio de Mesquita Filho

, São José do Rio Preto, 2021.

GOMES, V. Z. *Algoritmo genético híbrido com meta-heurísticas e aprendizado por reforço para alinhamento múltiplo de sequências*. 77 p. Tese (Tese (Doutorado em Ciência da Computação)) — Universidade Estadual Paulista "Júlio de Mesquita Filho

, São José do Rio Preto, 2025.

HOGEWEG, P.; HESPER, B. The alignment of sets of sequences and the construction of phyletic trees: an integrated method. *Journal of Molecular Evolution*, Springer, v. 20, n. 2, p. 175–186, 1984.

JR, J. W. H.; SKADRON, K. Accelerated warmup for sampled microarchitecture simulation. *ACM Transactions on Architecture and Code Optimization (TACO)*, ACM New York, NY, USA, v. 2, n. 1, p. 78–108, 2005.

JUMPER, J. et al. Highly accurate protein structure prediction with alphafold. *Nature*, v. 596, n. 7873, p. 583–589, Aug 2021. Disponível em: <https://doi.org/10.1038/s41586-021-03819-2>.

KANEHISA, M.; BORK, P. Bioinformatics in the post-sequence era. *Nature genetics*, Nature Publishing Group, v. 33, n. 3, p. 305–310, 2003.

KARAS, M. B.; HERMEL, E. d. E. S.; GÜLLICH, R. I. d. C. Modalidades didáticas: O ensino de virologia na educação básica. *Revista de Ensino de Biologia da SBEnBio*, v. 11, n. 1, p. 73387, oct 2018.

LEE, Z.-J. et al. Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. *Applied Soft Computing*, Elsevier, v. 8, n. 1, p. 55–78, 2008.

LEMOS, M.; CASANOVA, M. A. Algoritmos para análise de sequências. *PUC-Rio Departamento de Informática*, 2023.

LIU, L.; ABBAS, A. Bioinformatics. *Wiley Encyclopedia of Biomedical Engineering*, John Wiley & Sons, Inc. Hoboken, NJ, USA, 2006.

MUZZEY, D.; EVANS, E. A.; LIEBER, C. Understanding the basics of ngs: from mechanism to variant calling. *Current genetic medicine reports*, Springer, v. 3, n. 4, p. 158–165, 2015.

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, Elsevier, v. 48, n. 3, p. 443–453, 1970.

NETO, S. R. da S. et al. Valeria: um aplicativo para auxiliar no diagnóstico diferencial de arboviroses. In: SBC. *Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia)*. [S.l.], 2023. p. 115–118.

NICKOLLS, J.; DALLY, W. J. The gpu computing era. *IEEE micro*, IEEE, v. 30, n. 2, p. 56–69, 2010.

NOTREDAME, C.; HIGGINS, D. G. SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Research*, Oxford University Press, v. 24, n. 8, p. 1515–1524, 1996.

NOTREDAME, C.; HIGGINS, D. G.; HERINGA, J. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, Elsevier, v. 302, n. 1, p. 205–217, 2000.

OLIVA, G. Bioinformática: perspectivas na medicina. *Gazeta Médica da Bahia*, v. 78, n. 1, 2008.

PANDEY, M. et al. The transformational role of gpu computing and deep learning in drug discovery. *Nature Machine Intelligence*, v. 4, p. 211–221, 03 2022.

PERRONE, J. V. S. C. Dados demográficos, clínicos e determinação da variante viral de sars-cov-2 em casos fatais de covid-19 na bahia. 2022.

PROSDóCIMI, F. et al. Bioinformática: Manual do usuário. *Biotecnologia Ciência Desenvolvimento*, Universidade Federal de Minas Gerais, n. 29, 2023.

PUJOL, F. Importancia de la genómica en enfermedades emergentes: Covid-19 y viruela símica. *Vitae*, n. 93-96, 2023.

RODRIGUES, G. M. Variantes de preocupação do sars-cov-2: identificação presuntiva por sequenciamento de sanger da região do domínio de ligação ao receptor do gene s para vigilância genômica viral no hospital de clínicas de porto alegre. 2023.

RODRIGUES, P. H. M. *Desenvolvimento de antígeno vacinal com possível atividade terapêutica contra o papilomavírus humano, utilizando a tecnologia de DNA recombinante*. Tese (Doutorado) — Universidade de São Paulo, 2024.

RUBIO-LARGO, Á.; VEGA-RODRÍGUEZ, M. A.; GONZÁLEZ-ÁLVAREZ, D. L. Hybrid multiobjective artificial bee colony for multiple sequence alignment. *Applied Soft Computing*, Elsevier, v. 41, p. 157–168, 2016.

RUCCI, E. et al. Swimm 2.0: enhanced smith-waterman on intel's multicore and manycore architectures based on avx-512 vector extensions. *International Journal of Parallel Programming*, v. 47, p. 296–316, 2019.

SANGER, F.; NICKLEN, S.; COULSON, A. R. Dna sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences of the USA*, v. 74, n. 12, p. 5463–5467, Dec 1977. Disponível em: <https://www.pnas.org/doi/10.1073/pnas.74.12.5463>.

SANTOS, J. A. et al. Modelagem de proteínas por homologia. *Qualis-Qualidade em Revista*, v. 25, n. 1, p. e10001, 2023.

SANTOS, W. F. et al. Sequenciamento de dna: métodos e aplicações. In: *Proceedings of Safety, Health and Environment World Congress*. [S.l.: s.n.], 2013. v. 13, p. 139–141.

SCHMIDT, B. et al. Cudasw++ 4.0: ultra-fast gpu-based smith–waterman protein sequence database search. *BMC bioinformatics*, Springer, v. 25, n. 1, p. 342, 2024.

SHEHAB, S.; KESHK, A.; MAHGOUB, H. Fast dynamic algorithm for sequence alignment based on bioinformatics. *International Journal of Computer Applications*, v. 37, p. 54–61, 01 2012.

SMITH, T. F.; WATERMAN, M. S. Identification of common molecular subsequences. *Journal of Molecular Biology*, v. 147, n. 1, p. 195–197, 1981.

SOUZA, L. C. d. *Nova proposta de representação de genoma viral aplicada na classificação do Sars-Cov-2 com aprendizagem profunda*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2022.

THOMPSON, J. D.; HIGGINS, D. G.; GIBSON, T. J. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, Oxford University Press, v. 22, n. 22, p. 4673–4680, 1994.

WANG, L.; JIANG, T. On the complexity of multiple sequence alignment. *Journal of computational biology*, v. 1, n. 4, p. 337–348, 1994.

ZIVANOV, J. et al. New tools for automated high-resolution cryo-em structure determination in relion-3. *eLife*, v. 7, p. e42166, Nov 2018. Disponível em: <https://doi.org/10.7554/eLife.42166>.