khazanah informatika

Jurnal Ilmu Komputer dan Informatika

Herbal Compound Screening with GPU Computation on ZINC Database through Similarity Comparison Approach

Refianto Damai Darmawan, Wisnu Ananta Kusuma*, Hendra Rahmawan
Computer Science Department
IPB University
Bogor, Indonesia
*ananta@apps.ipb.ac.id

Abstract-Covid-19 is a global pandemic that drives many researcher strive to look for its solution, especially in the field of health, medicine, and total countermeasures. Early screening with in-silico processes is crucial to minimize the search space of the potential drugs to cure a disease. This research aims to find potential drugs of covid-19 disease in ZINC database to be further investigated through in-vitro method. About 997.402.117 chemical compounds are searched about its similarity to some of confirmed drugs to combat coronavirus. Sequential computation would take months to accomplish this task. General programming graphic processing unit approach is used to implement similarity comparison algorithm in parallel, in order to speed up the process. The result of this study shows the parallel algorithm implementation can speed-up the computation process up to 55 times faster, and also that some of the chemical compounds have high similarity score and can be found in nature.

Keywords: covid-19, GPU programming, parallel programming, similarity comparison

Article info: *submitted November 19, 2021, revised January 22, 2022, accepted February 24, 2022*

1. Introduction

a. Background

Covid-19 is a disease that is so phenomenal in 2020. This disease is caused by infection with the SARS COV-2 virus. Due to the nature of this virus that spreads quickly and is assisted by easy access for humans to carry out cross-country transportation, the problem of local viruses in China has become a global pandemic that has an impact on all countries in the world. During 12 to 18 July, 32 out of 34 provinces in Indonesia reported an increase in cases of which 17 provinces experienced an increase of 50% or more [1]. Researchers around the world are working on developing vaccines and drugs for COVID-19 (coronavirus disease). The way this virus enters the human body is through the ACE-2 receptor. The ligand referred to in the picture is a molecule that binds to another molecule, in this case the spike protein of the corona virus with the ACE-receptor.

The human need for drugs and means of prevention has prompted researchers to conduct research to find alternative drugs and vaccines for the corona virus. One alternative developed is the use of herbal plants to prevent and treat this disease [2]. The drug repurposing strategy is very useful considering that the conventional drug discovery process takes a long time. Drug repurposing is done by finding new benefits or efficacy of drug compounds that have been registered. Drug repurposing is usually done by analyzing the interaction of drug compounds with related proteins of a disease, then predicting new drug-target interactions that have not been known before [3].

The herbal medicine approach, or commonly called herbal medicine, is considered very useful for the prevention and treatment of COVID-19 disease due to several factors, namely the level of drug availability, drug safety, and the level of trust of the Indonesian people. As a traditional medicine made from plants, the level of availability of herbal medicines in Indonesia is so abundant, especially after being declared as one of the countries with a very wide and large plant biodiversity [4]. The safety of drugs from herbs or herbal plants has been tested from time to time because this method has been used for generations in traditional societies. The level of Indonesian people's trust in herbal medicine is also quite high, considering that more and more people are using herbs as an alternative treatment for various diseases.

The large amount of compound data available makes the sequential similarity search process take a very long

time, thus requiring a more efficient approach. In addition to the use of adequate hardware, computing speed is also affected by the algorithm or how the computer works. The concept of parallelization and the use of Graphics Processing Unit (GPU) to speed up calculations make computation time faster than the Central Processing Unit (CPU) for certain types of computing [5]. The search for the similarity of these compounds can use the parallelization concept provided by the GPU to speed up the process. In [5], the GPU-assisted version of Support Vector Machine (SVM) is developed to significantly decrease the processing time of SVM training for large scale training data. The result showed that the use of GPU is proven to be significantly decrease the training time. The bigger the dataset, the more training time reduction it gets from using GPU.

Parallel drug-target interaction (DTI) research has been carried out using several schemes, including breadth-first search (BFS) [6], molecular docking with GPU [7], and BINDSURF which is a virtual screening methodology to find a protein binding site for a ligand [8]. In [6], BFS is used to predict drug-target interaction in a graph and is optimized by parallelization using CUDA, which gained a speed-up of 51.33 times by using 4 threads. In [7], a novel molecular docking approach is proposed and optimized by using heterogeneous implementation based on multicore CPUs and multiple GPUs. The result shows that this novel approach is able to perform blind docking simulations in a scenario where the two prominent docking programs, i.e., AutoDock 4 and AutoDock Vina, are not able to perform. In addition to that, the average real-mean-square deviation (RMSD) score of the proposed method is lower than the average RMSD score of the other two. In [8], a virtual screening method is presented to find new hotspot, an area in a protein where ligands might interact with. The GPU parallelism is used to allow fast processing of large ligand database.

This study aims to find out the potential of GPU in the search for compound similarity by utilizing its parallelization potential. By knowing GPU performance in the search for this medicinal compound, it is hoped that it can be a reference for the next in-silico research. Aside from that, this study also aims to find the potential of herbal compounds that exist in nature as Covid-19 drugs, through searching for similarity with several existing drugs. The herbal compounds identified as similar to the Covid-19 drugs can be carried out by further in-vitro research for verification, and if scientifically proven they will be able to assist the public in finding alternative drugs to deal with the Covid-19 pandemic.

b. Problem Formulation

Problem formulation in this research is:

a) How to apply a similarity comparison algorithm to molecular compound data with GPU computing to determine the level of similarity?

b) What are the herbal plants that have a high potential to become drug candidates for the COVID-19 disease?

c. Aims

Aims of this research is:

a) Design, implement, and evaluate parallel computing solutions similarity comparison using GPU.

c) Finding candidate compounds that have the potential to prevent or treat Covid-19.

d. Benefit

The results of this study are expected to provide benefits to improve the quality and speed of the drug discovery and drug repurposing process in the world of herbal medicine or herbal medicines so that in the end it can help the community in overcoming relatively new diseases at affordable prices.

e. Research Scope

The scope of this research is to look for compound data in-silico without being accompanied by in-vitro and in-vivo tests. The data sought is limited to the compound SMILES (simplified molecular input line entry system) string information.

2. Literature Review

a. Ligand-based Compound Screening

Virtual screening is a chemoinformatics technology designed to evaluate a large number of compounds computationally, with the aim of quickly identifying the desired structure so that it can be submitted as a bioassay [2]. Traditionally, the screening process is carried out through high-throughput screening (HTS), which is testing several compounds in bulk to find compounds that hit (interact) with the target protein. However, post-HTS analyzes are often disrupted [9] by the presence of protein-reactive compounds [10] or optically interfering components, which are the result of sample degradation from biochemical assays [11], or the tendency of chemicals to conduct aggregation [12]. To overcome the weakness of the HTS, virtual screening was developed in order to obtain more accurate screening results.

In this study, the approach used is ligand-based compound screening, namely the selection of compounds based on the level of similarity (similarity) of a compound that has successfully bound to the desired ligand (active protein region).

b. Herbal Compound

Herbal medicine is defined as a collection of therapeutic experiences from generation to generation by traditional healers over hundreds of years [13]. Most of the sources of herbal medicines are plants, so that in their development herbal medicines are identical to medicines

derived from plants. Because herbal medicines are used by people based on people's habits, the scientific evidence they have is not strong enough. Further research on the real efficacy needs to be done in the process of using this herbal medicine in order to have strong evidence that the drug is safe for human use. To be accepted as a viable alternative to modern medicine, rigorous scientific and clinical validation methods must also be carried out to prove the safety and effectiveness of an herbal product [14].

In this study, the herbal compounds referred to refer to all active substances found in plants and have been used as medicine for generations in Indonesian society. One example that can be mentioned is curcumin in turmeric which is commonly used to relieve inflammation because it is anti-inflammatory [15].

c. Graphic Processing Unit

Graphic Processing Unit (GPU) is an electronic circuit hardware designed to manipulate memory quickly to accelerate image creation in a frame buffer that is intended as output on a display screen [16]. Although the original purpose of GPUs was to improve graphics performance, researchers often use them for the purpose of accelerating data processing. This happens because the GPU has many computational cores that can be used for parallel computing processes. The thing that users need to prepare is how to separate the data so that the GPU can work on it in parallel, then combine the results of the calculations so that there are no errors and the results are valid.

In Figure 1, you can see the differences in the architecture of the Central Processing Unit (CPU) and GPU. In the CPU architecture drawing, it can be seen that quite a lot of space is used for the control unit and cache. This makes sense because the CPU will receive a lot of data and commands that tend to be unique for each data to be processed, so it needs to be accommodated with an adequate cache and control unit. On the GPU side, it can be seen that the allocation of space for cache and control unit is relatively small and minimal, and mostly consists of relatively small but large number of arithmetic and logic units (ALU). This happens because the initial purpose of the GPU is to improve computer performance in processing graphic data so that it is processed faster so that it can appear on the screen faster. And this is achieved by increasing the number of ALU data processing units because the data processed is quite large and the instructions for processing the data are in the form of simple arithmetic, such as adding and subtracting times [17].

Figure 1. CPU and GPU architecture design illustration

Departing from the analogy of graphics processing, where each pixel will be processed in parallel, the GPU has many threads that are used as a place to process data. A collection of threads, called a block, has a shared cache or memory. A collection of blocks in the same place will form a grid [18]. On the GPU, the processing element used to process data is a thread. These abstractions will facilitate parallel programming when implementing research.

d. ZINC Database

ZINC is a commercially available molecular database. Basically, the molecular data contained in this database is in the form of a simplified molecular input line entry system (SMILES), but in its development, two-dimensional and three-dimensional representations of molecules are also available [19]. The ZINC database is often used in research to find ligands, which are ions or molecules that can attach to metal atoms by covalent bonds. These ligands are often used to find a cure for a disease or virus by disrupting the life cycle of the virus, or directly destroying the virus structurally. To find out the form of data from the ZINC database, please see Figure 2.

zinc_id	smiles
ZINC000245189325	O=P(=O)O
ZINC000029747110	COc1cccc2c1[C@@H]1CN(CCCCn3c(O)nc4c(sc5ncc(-c6ccccc6)nc54)c3=O)C[C@@H]1CO2
ZINC000137550338	CN(C)c1cc(CNCC(C)(C)C)c(O)c2c1C[C@H]1C[C@H]3[C@H](N(C)C)C(=O)[C@H](C(N)=O)C(=O)[C@@]3(O)C(=O)[C@H]1C2=O
ZINC000137550409	CN(C)c1cc(CNCC(C)(C)C)c(O)c2c1C[C@H]1C[C@H]3[C@H](N(C)C)C(=O)[C@@H](C(N)=O)C(=O)[C@@]3(O)C(=O)[C@@H]1C2=O
ZINC000137550489	CN(C)c1cc(CNCC(C)(C)C)c(O)c2c1C[C@H]1C[C@H]3[C@H](N(C)C)C(=O)[C@@H](C(N)=O)C(=O)[C@@]3(O)C(=O)[C@H]1C2=O
ZINC000005161047	Cc1ccc2c(n1)Oc1ccc([C@@H](C)C(=O)OCC(=O)N(C)C)cc1C2
ZINC000137550260	CN(C)c1cc(CNCC(C)(C)C)c(O)c2c1C[C@H]1C[C@H]3[C@H](N(C)C)C(=O)[C@H](C(N)=O)C(=O)[C@@]3(O)C(=O)[C@@H]1C2=O

Figure 2. Sample file download from ZINC database

e. Similarity Comparison Algorithm

Similarity comparison algorithm is an algorithm to determine the similarity between two data sequences. In bioinformatics, this algorithm has been found and used for quite a long time, namely since the 20th century [20]. The use and implementation of this algorithm is also wide, and depends on the type of data to be compared, such as protein sequence data, fingerprint data, binary data, and others. In this study, the data being compared is binary data in the form of fingerprint output from compound molecules.

Broadly speaking, the comparison of binary similarity carried out in this study aims to provide similarity values for two different binary strings [21]. For example, the binary strings being compared are 0110 and 1010. Each index in the string indicates the presence or absence of a

feature in the referenced compound. Tanimoto's algorithm will find the number of digits of 1 in both strings that are at the same index (in this example it is the third index), and divide it by the number of digits of 1 in both strings at different indices (in this example, there is a value of 1 at the first index, second, and third). Tanimoto's algorithm will give 1/3 value for both binary strings above. In other words, Tanimoto's algorithm will divide the number of characteristics that are the same in both compounds by the total number of characteristics that exist in both compounds, whether only one compound has one or both.

Besides Tanimoto, there are several other algorithms to determine the value of binary similarity, namely Dice and Cosine. Dice's algorithm multiplies the number of traits that are the same in both compounds by 2, and then divides by the number of traits that exist in both compounds. The cosine algorithm will divide the number of features that are the same in both compounds by the root value of the product of the number of features that only one compound has. This study uses the Tanimoto algorithm because it is quite simple and has proven to be suitable for use as a basis for calculating the similarity of chemical compounds according to [21].

3. Methods

a. Research Stages

Figure 3 describes the stages and research methods used. As initial data, data on ZINC compounds were collected in the biogenic sub-group which amounted to 308.035 compounds. The fingerprint calculations used are MACCS and PubChem, and are carried out using CPU sequential and CPU parallel methods. After the fingerprint data is formed, parameter tuning is carried out on the GPU parallelization scheme, namely determining the block size and determining the number of streams. After the block size and the appropriate number of streams are determined, a similarity comparison process is carried out which is implemented using two methods, namely GPU sequential and GPU parallel. After calculating the time, the evaluation of the two methods is carried out by calculating the speed-up value.

After the best model was found, the overall ZINC data, which amounted to 997,402,117 compounds and the PubChem fingerprint was calculated, which was applied to the model to find compounds similar to Covid-19 drug compounds.

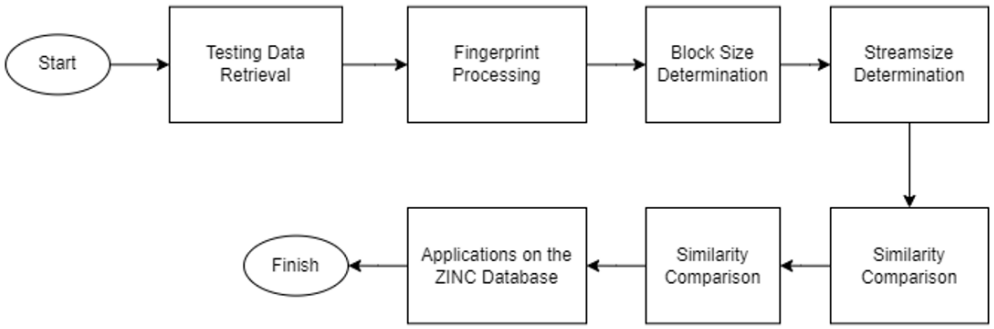

Figure 3. Flowchart of research stages

b. Research Data

The data used and analyzed in this study are all compound SMILES data in the ZINC database. Due to the large amount of data, the researchers took some of the ZINC data in the biogenic subset of 308.035 compounds as the basis for developing the algorithm. The data on these compounds were sought for the level of similarity to eight Covid-19 drug compounds that have been approved or are considered strong candidates in the medical world. To facilitate the search, a list of medicinal compounds used as a reference can be seen in Table 1.

Table 1. List of medicinal compounds used as a reference

Compound Name	How It Works	Reference
Remdesivir	Stops the replication of coronavirus RNA[a] inside the host cell.	[22]
Favipiravir	RNA-dependent RNA-polymerase inhibitors in common cold viruses.	[23]
Lopinavir	Antiretroviral agents, protease inhibitors.	[24]
Hydroxi-chloroquine	Causes alkalization in cells, preventing the acidization needed by viruses for replication.	[25]
Chloro-quine	Causes alkalization in cells, preventing the acidization needed by viruses for replication.	[25]
Nitazox-anide	Suppress inflammation during a cytokine storm.	[26]
Oseltamivir	Inhibitors on the corona virus 3CLpro protein.	[27]

[a]Ribonucleic Acid, carrier of genetic information in virus.

c. Fingerprint Processing

As can be seen in Figure 4, the data obtained from the ZINC database consists of zinc_id, which is a unique code for indexing each compound, and

SMILES. SMILES is a form of writing compounds in the form of strings so that they can be represented concisely and clearly. However, this representation cannot be used as a reference that a compound is similar or not, because SMILES is just a simple string form of a compound.

This SMILES representation needs to be converted into fingerprint form which is a collection of compound representations in binary form that is used to determine the level of similarity of a compound with other compounds. The use of fingerprints as data representation of a compound makes similarity comparison results more accurate because we judge from the properties brought by the structure of the compound itself.

This study uses two types of SMILES fingerprints that are commonly used, namely MACCS and PubChem fingerprints. MACCS encodes a compound into 166 binary bits, while PubChem fingerprint will encode a compound into 881 binary bits. Each binary bit present in MACCS and PubChem fingerprints is a molecular identifier that chemists commonly use to group compounds. Some examples of the characteristics used are the number of C atoms in the compound which is more than 4, as well as the presence of a cis/trans structure.

```
ιead(df1)
        zinc_id                                                                                                    smi
:INC000017653949 COC(=O)[C@H]1[C@H](OC)[C@@H](OC(=O)c2cc(OC)c(OC)c(OC)c2)C[C@@H]2CN3CCc4c([nH]c5cc(OC)ccc45)[C@H]3C[C@H
:INC000031502642                     COC(=O)C1=CO[C@@H](O[C@@H]2O[C@H](CO)[C@@H](O)[C@H](O)[C@H]2O)[C@H]2[C@@H]1C=C[C@]2(O
:INC000006018601                                             C[C@@H]1CC[C@@]2(C)[C@H](C)/C(=C\\COC(=O)/C=C/c3ccc(O)c(O)c3)CC[C@@]
:INC000017654726                               COc1cc2c(c(OC)c1OC)-c1c(cc3c(c1OC)OCO3)C[C@H](C)[C@@](C)(O)[C@@H]2OC(=O)c1ccc
:INC000001721658                                                                                   CC1(C)[C@H]2CCC(=O)[C@@H]
:INC000032296250                      CC1=CC(=O)[C@@]2(C(C)C)C(=O)C3=C([C@H](O)[C@H]4OC(=O)[C@@]35CCCC(C)(C)[C@H]45)[C@@H
```

Figure 4. Data downloaded from ZINC database in Rstudio environment

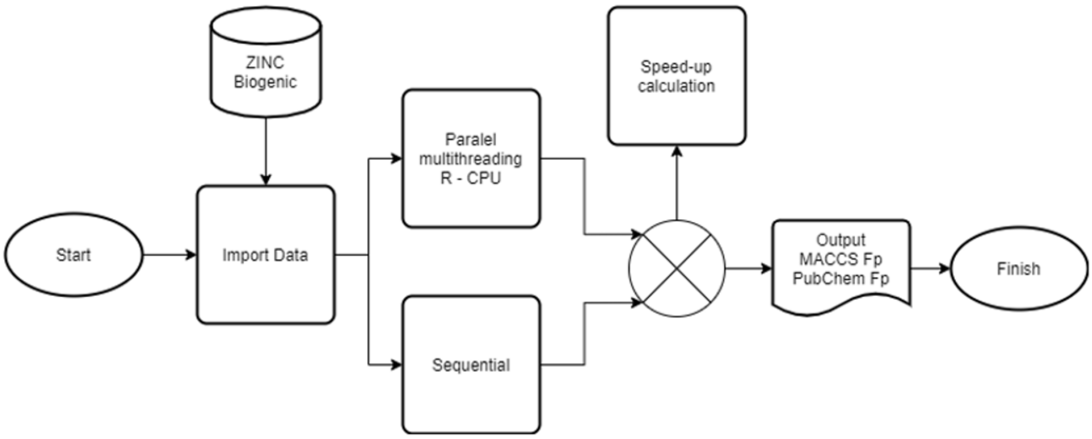

Figure 5. Fingerprint processing flow chart

```
zinc_id","pubchem"
ZINC000000538065","000000000111101110110000000000000000000100000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
1011000000000000000000000000000000001111000110000010000000000000000000000000000000
0000000000000000000000000000000011101000000000000000000000000001111000000010000010
0000000000000000000000000000100000001110111000011001001100100110001100111101011
0001110000110000010100000000000101010100000000000001001100110001001110100000000001
1001010001000000010110010000110100111101000000000111111001000000000000001111001
1110100010001000111001101110001000101100010111111011100110111001011100111000001
0001110110011001111000100111101100011101000000000111100100000100000000000000000
0000000000000000000000000000000000000000010000000000000000000000000000000000000000
0000000000000000000000001000000000000000000000000000000000000000000000000000000000
000000000"
```

Figure 6. PubChem fingerprint representation example

In general, the flowchart of the fingerprint processing process is depicted in Figure 6. This process begins by importing a subset of ZINC data into the RStudio software, then performs two kinds of fingerprint processing, namely CPU sequential and CPU parallel. The sequential process will process MACCS and PubChem fingerprints one by one. The parallel process will divide this fingerprint processing task into all available CPU cores, i.e. 32 cores. The parallelization process is carried out with the CPU, because it is more practical and already supports the required library, i.e. rcdk.

After the fingerprint processing is complete, then the speed-up value is calculated as a result of sequential and parallel processing time. The output produced is a MACCS fingerprint of 166 bits and a PubChem fingerprint of 881 fingerprints. This output will be used in the next process, namely calculating the similarity or similarity using CUDA. The flowchart of this process can be seen in Figure 5. An example of the resulting fingerprint can be seen in Figure 6.

```
READ biogenic data subset
FOR every row of SMILES representation of a
compound
   DO convert to fingerprint representation
ENDFOR
```

Figure 7. Pseudocode for fingerprint processing CPU sequential algorithm

```
READ biogenic data subset
DO divide data according to number of CPU thread
FOR every CPU thread available
     FOR every row of SMILES representation of a
     compound
          DO convert to fingerprint
          representation
     ENDFOR
ENDFOR
DO concatenate fingerprint result in order
```

Figure 8. Pseudocode for parallel CPU fingerprint processing algorithm

Figures 7 and 8 show the algorithm used to process fingerprints, both MACCS and PubChem fingerprints. The parallelization scheme used is CPU parallelization with Single Instruction Multiple Data (SIMD).

d. Block Size Determination

Determining the block size is important to determine the optimal value of threads per block that will be used in the GPU parallelization process. Figure 9 shows the process of determining the most optimal block size to use. Since the CUDA architecture has a warp size of 32, the block size will also be determined in multiples of 32, up to a maximum value of 1024 threads per block.

```
FOR every iteration from 1 to 7
     FOR every block size ranged from 32 to 1024
     in multiples of 32
          DO time calculation of the similarity
          comparison on PubChem fingerprint
          biogenic data with 7 PubChem
          fingerprint covid drugs
     ENDFOR
ENDFOR
CALCULATE the average value of each block size
on 7 iterations
DETERMINE block size with shortest average time
DO concatenate fingerprint result in order
```

Figure 9. Pseudocode for block size determination algorithm

e. Streamsize Determination

Determining the streamsize is needed to find out

```
FOR each iteration value from 1 to 7
      FOR any number of streams that are 1 to
100 in multiples of 1
           DO calculation of the similarity
           comparison between PubChem
           fingerprint biogenic data with 7
           pubChem fingerprint covid drugs
      END FOR
END FOR
CALCULATE the average value of each number of
streams in 7 iterations
DETERMINE the number of streams with the fastest
average time
```

Figure 10. Pseudocode for streamsize determination algorithm

f. Similarity Comparison

1) Tanimoto Similarity

Tanimoto similarity algorithms will compare each bit in the same location. This is done considering that fingerprints at the same position show the same compound markers, so we can see whether or not a pair of compounds from these markers is similar.

For example, in Figure 11 there is one pair of compound fingerprints being compared. Tanimoto similarity is calculated by counting the number of all bits in a position where the two compounds have the same value, which is one, then divided by the number of all bit positions where one of the two compounds is worth one. In the example above, there are two positions where the two compounds have the same value of one, namely the fourth and fifth positions. And there are six positions where one of the two compounds is worth one, namely positions one, three, four, five, six, and seven. So that the similarity value of the two compounds is two-sixth, or in other words one-third.

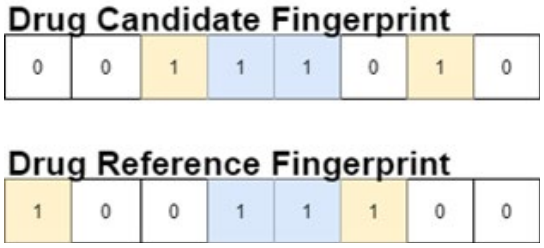

Figure 11. Illustration of Tanimoto similarity calculation

2) Sequential Algorithm

The sequential algorithm takes one candidate compound data represented by one data line, then calculates its similarity with several drug compounds that have been selected, and stores the results in a table that is sized according to the number of candidate compounds and drug compounds.

In the illustration in Figure 12, as well as the pseudocode in Figure 13, we take one fingerprint data on the top row, then compare it with the three existing

drug fingerprints. The results of the Tanimoto similarity calculation are written in the top row Similarity Results table, so that each row of the results table represents the candidate compound, and each column represents the drug compound being compared. The table of Similarity Results in the first row and third column shows the Tanimoto similarity value between the first candidate compound and the third drug compound.

As the name implies, this algorithm is performed sequentially in a CPU thread, so the processing time depends on the CPU's ability to process data. This implementation is written in a Python language program using the pandas library to perform data import, data export, and dataframe management, as well as the NumPy library to perform operations on two-dimensional arrays.

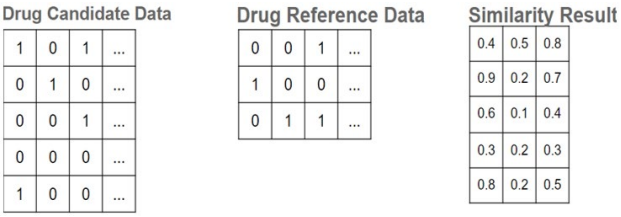

Figure 12. Representation of the similarity comparison process

```
FOR each line fingerprint representation of drug
candidate compounds
      FOR each line fingerprint representation
      on Covid-19 drug reference compounds
            DO comparison of similarity
      ENDOR
ENDORDO concatenate fingerprint result in order
```

Figure 13. Pseudocode for sequential algorithm for similarity comparison on CPU

3) Parallelization Algortihm

Parallelization of the similarity comparison process is carried out by utilizing the Nvidia CUDA General Purpose GPU (GPGPU) as a processor. Each thread in the GPU is used to process one candidate compound with the eight drug compounds. After calculating the similarity value, this GPU writes the value to the result table, like the sequential process above. This term is known as single instruction, multiple data (SIMD).

Figure 14 also shows that apart from dividing each compound into one thread, multistreaming is also carried out in this process, namely dividing the parallel process into several different streams. This is achieved by dividing the compound data equally into each stream, and copying the same drug reference data to each stream to be used, so that each stream does not need to communicate with other streams and can run optimally.

This mechanism can speed up the parallelization process because each stream has its own clock, allowing these processes to run asynchronously. Of course, this process only occurs until the data is written into the similarity table. At the end of the process, each stream collects a table of results and rewrites it in a table of similarity results sequentially, starting from the first stream to the last. The writing of these results is carried out in synchronization, so that there is no overlap, and the order of the resulting data tables is in accordance with the order of the compounds used. This algorithm is briefly described in the pseudocode in Figure 15.

In GPU parallelization abstraction, apart from being known as a thread that does real work, there is also a term called a block which is a collection of threads that are in the same container. Programmers need to determine the appropriate block size so as to achieve maximum performance gain. Therefore, in this study, various parallelization schemes with different block sizes were used, then repeated seven times, so as to find a block size that could process the training data optimally. The block size values used are in the range of 32, 64, 96, and so on up to 1024. This is because the maximum number of threads in one block on the CUDA architecture is only up to 1024 threads per block, and the warp size on the CUDA architecture is 32, so that to achieve optimal efficiency it is necessary to have a block size that has multiples of 32.

As mentioned above, the multistreaming mechanism is implemented so that each parallel process runs asynchronously, so it can run faster. This study also calculates similarity with a number of different streams, then repeated seven times, so that researchers can determine the best number of streams to use in parallelizing the entire ZINC database. The value of the number of streams used is in the range of one to one hundred streams.

This parallel similarity comparison process is written in the Python programming language as the basis, with the help of the PyCUDA library which is used as programming code that occurs on the GPU. On each GPU thread, the program code is written in C++, according to the needs of the CUDA library in order to execute its commands.

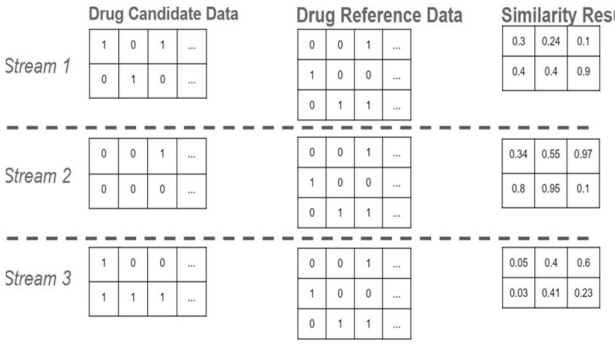

Figure 14. Representation of the parallel comparison process

```
DO stream creation according to the optimal
number of streams
DO duplicate Covid-19 drug fingerprint data
DO distribution of drug candidate compound data
to each stream according to the number of
existing streams
FOR every stream on GPU
      FOR each GPU thread that calculates
      similarity values line fingerprint
      representation on drug candidate compounds
            FOR each row fingerprint
            representation of the reference drug
            data
                  DO comparison of similarity
            ENDOR
      ENDOR
ENDOR
DO concatenation of similarity results in order
```

Figure 15. Pseudocode for parallel GPU similarity comparison algorithm

g. Algorithm Evaluation

Parallel algorithms and sequential algorithms are compared by means of calculating the speed-up value. The speed-up value is the ratio of the time it takes to run the sequential algorithm to the time it takes to run the parallel algorithm. For example, if the sequential algorithm takes 9 seconds and the parallel algorithm takes 5 seconds, then the speed-up value obtained is 1.8.

$$speed\text{-}up = t_{sequential} / t_{parallel} \quad (1)$$

Equation (1) is a general formula for calculating speed-up values, where $t_{sequential}$ is the time it takes to do something sequentially, and $t_{parallel}$ is the time it takes to do something in parallel. The greater the difference in sequential and parallel time, the higher the speed-up value, with a note that the sequential time is longer than the parallel time.

h. Application on the ZINC Database

After obtaining a good parallelization model from the biogenic subset ZINC training data, the model was applied to all datasets in the ZINC database. This parallelization process takes a long time so that by implementing parallelization, data processing time can be accelerated. The results of processing this entire database show information about several potential compounds that are similar to Covid-19 drug compounds. This study uses the PubChem fingerprint to process the entire ZINC database because it has more descriptors, so it can identify similar compounds more accurately. For comparison, the training data used were 308,035 compounds, while the ZINC database totaled 997,402,117 compounds.

i. Development Environment

The software and hardware specifications used in this study are as follows:

Device type	: Desktop PC
Operation System	: Xubuntu 18.04
Processor	: Intel Xeon Silver 4110 2,2 GHz
Memory	: 64 GB
GPU	: NVIDIA RTX 2080
Storage Drive	: SSD 512 GB, HDD 7 TB
Programming language	: C++, Python, dan R
Library support	: PyCUDA dan rcdk
Software	: RStudio dan Visual Studio Code

4. Results and Discussion

a. Fingerprint Processing

Figure 16 and Figure 18 show the difference in processing time of the sequential and parallel algorithms for both MACCS and PubChem fingerprints. The graphs shown in Figure 17 and Figure 19 show the speed-up values obtained for the conversion process from the SMILES representation to MACCS and PubChem fingerprints. Overall, the speed-up value obtained is quite large and significant. This shows that the use of parallel processing with multithreaded CPUs to process fingerprints can have a significant impact on processing time.

In Figure 17, there is a very significant increase in the speed-up value of the number of drug candidates. In the number of candidate drug compounds, which amounted to a thousand and under, an insignificant increase was seen. The increase in the speed-up value is starting to be large and can be seen in the amount of data as much as five thousand and above. By looking at this graph, it can be seen that the scalability potential of MACCS fingerprint processing is still very high, or in other words, the -speed-up value can still increase again as the number of candidate compounds processed increases. The graph also shows that further research is needed on the potential limitations of parallel processing for MACCS fingerprint processing to get the optimal speed-up value. So far, the highest speed-up value of 27.73 was obtained from a total of 300,000 drug candidates. In other words, if we parallel processing MACCS fingerprints on 300,000 compound data, it will be 27 times faster than if we process them sequentially. Taking into account the number of processor threads, which are 32, the efficiency of the MACCS parallelization process with 300,000 drug candidates is 86.66%.

In Figure 19 it can be seen that the speed-up value that occurs in the PubChem fingerprint processing of 15.27 tends to be smaller when compared to the MACCS fingerprint which can reach a value of 27. Taking into account the number of threads, an efficiency value of 47.72% is also obtained. There was a significant increase in the number of candidates from ten, fifty, one hundred, and five hundred. For the number of candidates of a thousand and above, the speed-up value looks increasingly sloping, until the values of 50,000, 100,000, and 300,000 appear to have gone up and down, so it is quite possible that the threshold has been reached in this area. The speed-up difference between MACCS and PubChem fingerprint processing is probably caused by the number of bits in MACCS and PubChem, namely MACCS with 166 bits and PubChem with 881 bits. Of course this will affect the work of each thread that processes the fingerprint, so the work done by a thread to process the PubChem fingerprint will be greater than the MACCS fingerprint. This also explains the speed-up value that tends not to increase in MACCS fingerprint processing for the number of candidate compounds of ten, fifty, one hundred, five hundred, and one thousand, because the time used to process fingerprints is shorter than the time used to divide the fingerprints raw data and collect processing data from each working thread.

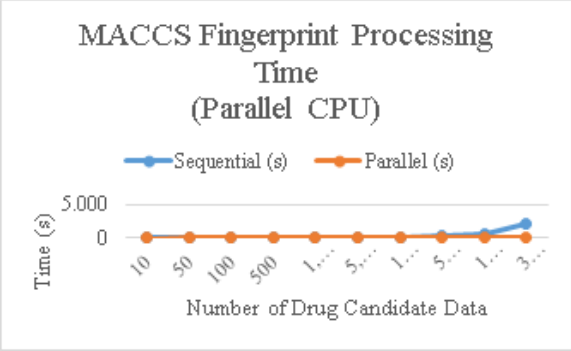

Figure 16. Graph of MACCS fingerprint processing time (Parallel CPU)

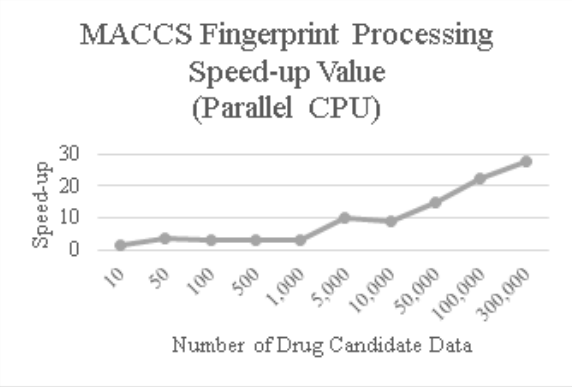

Figure 17. Graph of MACCS fingerprint processing speed-up value (parallel CPU)

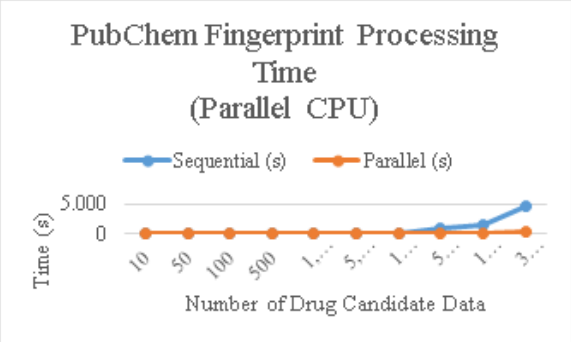

Figure 18. Graph of PubChem fingerprint processing time (parallel CPU)

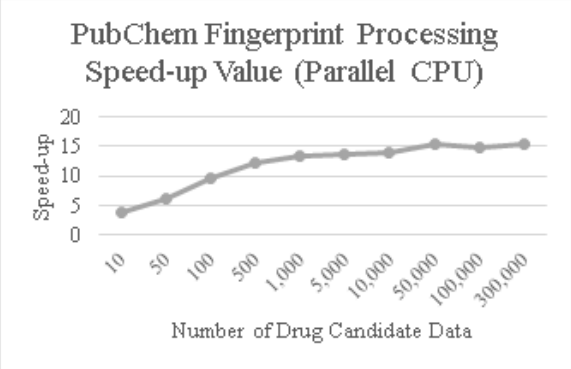

Figure 19. Graph of PubChem fingerprint processing speed-up value (parallel CPU)

b. Block Size Determination

Figure 20 shows that the run times for various block sizes are dynamic and have a global minimum of around 600 threads per block. The dark yellow line shows the average processing time of seven replicates. After looking for the least average value, we get a block size of 640 threads per block with an average time of 1.176 seconds.

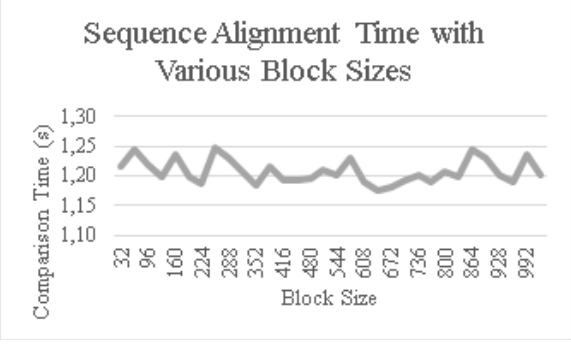

Figure 20. Graph of block size value with comparison time on the similarity comparison algorithm

c. Streamsize Determination

The average value indicated by the dark yellow line in Figure 21 shows the wave-like oscillation or looping. It can be seen in the number of flows from one to five, the processing time tends to decrease, then continues at the

number of flows above five which slowly shows an increase. The lowest value refers to the number of flows of five with a processing time of 0.828 seconds. This shows that the use of the number of streams less than five is included in a non-optimal state, because the number of streams is not proportional to the amount of data processed. A value of more than five, which indicates an increase in processing time, indicates that the number of streams is too large for the size of the data being processed, so the computer does the unnecessary work of creating and allocating data divisions into these additional streams.

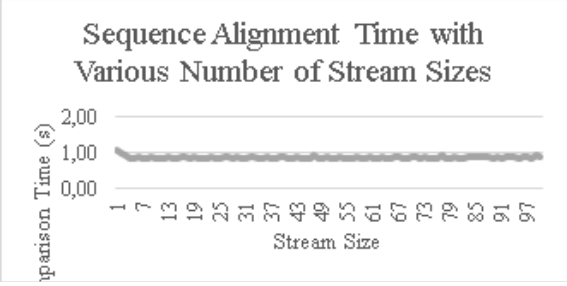

Figure 21. Graph of the amount of stream size against the comparison time on the similarity comparison algorithm

d. Similarity Comparison

Figure 22 and Figure 24 show the time difference between the sequential and parallel similarity comparison processes, for both MACCS and PubChem fingerprints. The graphs shown in Figure 23 and Figure 25 show that the sequential and parallel comparisons of the similarity comparison algorithms, for both MACCS and PubChem fingerprints, result in significant speed-up values. For MACCS fingerprint, the highest speed-up value is at 19.05 which is in the number of drug candidates of 300,000, while for PubChem fingerprint, the highest speed-up value is at 55.51 which is in the number of drug candidates of 100,000. This difference in speed-up values indicates that the tasks that can be parallelized on the PubChem fingerprint tend to be more, given the number of bits that reach 881.

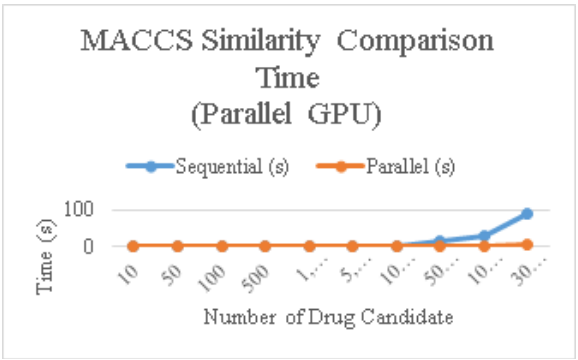

Figure 22. Graph of the similarity comparison processing time with the MACCS fingerprint (parallel GPU)

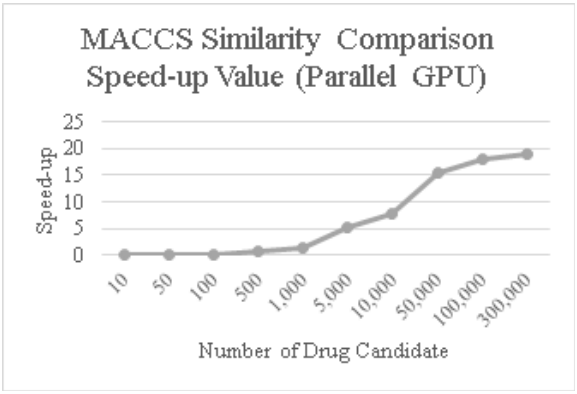

Figure 23. Graph of speed-up value for comparison of similarity with MACCS fingerprint (parallel GPU)

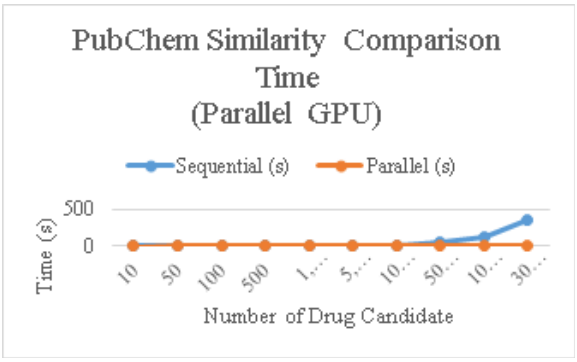

Figure 24. Graph of the similarity comparison processing time with the PubChem fingerprint (parallel GPU)

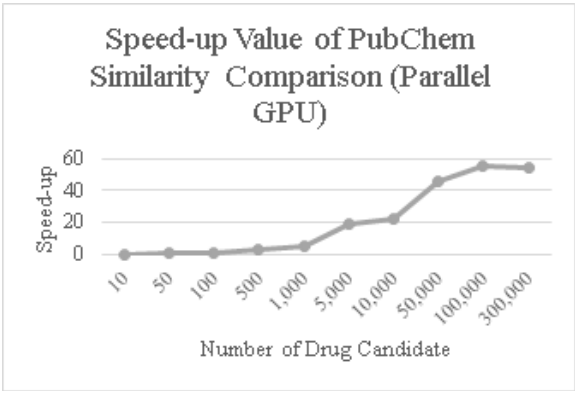

Figure 25. Graph of speed-up value comparing similarity with PubChem fingerprint (GPU parallel)

e. Application Result on ZINC Database

After applying the similarity comparison algorithm in parallel to the entire ZINC database, several compounds have the potential to be drug candidates. Most of the compound data that has a high similarity value are included in the unnamed compound category, so further identification cannot be done. In summary, the results of the similarity search can be seen in Table 2.

Table 2. Caption of my table (inf_tableHead)

Drug Name	*Siimilarity Value*	*Drug Candidate ID*	*Drug/Compound Candidate Name*	*Natural Source*
Remde-sivir	0.7799	ZINC- 29134440	Brivanib Alaninate	-
	0.7567	ZINC- 13684256	Brivanib	-
	0.7391	ZINC- 53147179	Puromycin	*Streptomyces alboninger* bacteria [28]
Favipi-ravir	0.7500	ZINC- 5116994	5-hydroxypy-razinamide	-
	0.7478	ZINC- 1081066	2-carboxy-pyrazine	-
	0.6850	ZINC- 500059	Pyrazine Carboxylic Acid Hydrazide	-
Lopi-navir	0.8214	ZINC- 49889244	Gliotide	*Gliocladium sp.* fungi [29]
	0.8214	ZINC- 95607016	Paecilodepsi-peptide C	*Paecilomyces cinnamomeus* fungi [30]
	0.8083	ZINC- 230078061	Adouetine Y	*Discaria Americana* tree bark [31]
Hydro-xichlo-roquine	1.000	ZINC- 1530654	Hydroxy-chloroquine	*Peruvia cinchona* tree bark [32]
	0.9934	ZINC- 1843038	Cletoquine	-
	0.9334	ZINC- 19144231	Chloroquine	*Peruvia cinchona* tree bark [32]
Chloro-quine	1.000	ZINC-19144231	Chloroquine	*Peruvia cinchona* tree bark [32]
	0.9929	ZINC- 1873617	Desethyl-chloroquine	-
	0.9858	ZINC- 6036375	Bidesethyl-chloroquine	-
Nitazo-xanide	0.8971	ZINC- 5924265	Tizoxanide	-
	0.8839	ZINC- 29124339	Tizoxanide Glucuronide	-
	0.7273	ZINC- 2257	Zolamine	-
Oselta-mivir	0.7593	ZINC- 13370140	Antillatoxin	*Lyngbya majuscule* [33]
	0.7265	ZINC-169367255	Aspochalasin J	*Aspergillus flavipes* [34]
	0.7248	ZINC-169290233	Cespitulactam J	*Cespitularia taeniata* [35]

Starting from the first drug, a candidate drug compound similar to remdesivir is puromycin, but this is not feasible for further research because of the low similarity value of 0.7391. The other two compounds cannot be found in natural sources, and their production can only be carried out in the laboratory.

The second drug, favipiravir, had no naturally-derived compounds in its top search results. It can also be seen that the most similar to favipiravir is 5-hydroxypyrazinamide which has a similarity level of 0.75, low enough to be considered similar.

The third drug, lopinavir, has several candidate compounds that are quite similar, namely gliotide, paecilopdepsipeptice C, and adouetine Y. These three compounds can be found sequentially in the fungus *Gliocladium sp.*, the fungus *Paecilomyces cinnamomeus*, and the bark of *Discaria americana*. It should also be noted that the similarity value of these three candidate compounds is not too high, which is in the range of 0.80 – 0.82.

For the fourth and fifth drug candidates, namely hydroxychloroquine and chloroquine, both can be found in the bark of the *Peruvia cinchona* plant. The thing to

note is that this plant is not native to Indonesia, so it will be difficult to find it.

The sixth drug, nitazoxanide, had no naturally-derived compounds in its top search results. Compounds such as tizoxanide, tizoxanide glucuronide, and zolamine can only be produced in the laboratory.

The seventh drug, namely oseltamivir, had similarity to natural sources in the similarity range of 0.70 to 0.75. 75% similarity was found in antillatoxin compounds, namely toxin compounds obtained from the marine cyanobacteria *Lyngbya majuscula*. 72% similarity was found in aspochalasin J and Cespitulactam B compounds, which were found in the fungus *Aspergillus flavipes* and the coral *Cespitularia taeniata*, respectively.

Finally, because the limitation of this research is to search for herbal ingredients, namely natural ingredients derived from plants, this search resulted in *Gliocladium sp.*, *Paecylomyces cinnamomeus*, *Discaria americana*, and *Peruvia cinchona*.

5. Conclusion and Suggestion

a. Conclusion

The similarity comparison algorithm can be applied to find the similarity value in molecular compound data in the form of SMILES by finding the descriptor in the form of a fingerprint, then comparing the two compound fingerprints with Tanimoto similarity. The use of GPU computing can increase the performance of this process up to 55 times faster.

There are several candidate herbal plants, and even the coronavirus drug compounds themselves can be found in nature and become candidates for Covid-19 drug compounds. Some of the herbal ingredients are difficult to find in Indonesia because they are from abroad.

b. Suggestion

Although this research focuses on herbal medicines, it does not rule out the possibility that non-herbal compounds obtained from the ZINC database screening can be a more practical alternative to become Covid-19 drugs. Researchers hope that the results of this study can be suggestions, input, and motivation to carry out further analysis in the laboratory to identify herbal ingredients that can be used to treat the corona virus.

In addition, researchers also hope that the results of research on the content of herbal compounds present in plants and organisms from Indonesia can be collected into a single database so that a comprehensive in silico analysis can be carried out. This suggestion arises from the difficulty of researchers in further identifying whether a compound is present in certain herbal plants in Indonesia.

Reference

[1] World Health Organization Indonesia, "Weekly Epidemiological Update on COVID-19," World Health Organization Indonesia, Jakarta, Indonesia, 23 Jul. 2021.

[2] C. G. Bologa, O. Ursu, and T. I. Oprea, "How to prepare a compound collection prior to virtual screening," *Methods in Molecular Biology,* vol. 1939, pp. 119–138, 2019.

[3] T. T. Ashburn and K. B. Thor, "Drug repositioning: Identifying and developing new uses for existing drugs," *Nature Reviews Drug Discovery*, vol. 3, no. 8, pp. 673–683, 2004.

[4] F. Medail and P. Quezel, "Biodiversity hotspots in the Mediterranean Basin: Setting global conservation priorities," *Conservation Biology*, vol. 13, no. 6, pp. 1510–1513, 1999.

[5] A. Athanasopoulos, A. Dimou, V. Mezaris, I. Kompatsiaris, Z. Wen, J. Shi, et al., "Performance of deep learning computation with TensorFlow software library in GPU-capable multi-core computing platforms," in *International Workshop on Image Analysis for Multimedia Service,* vol. 19, pp. 240–242, 2017.

[6] A. Reinaldo, W. A. Kusuma, H. Rahmawan, and Y. Herdiyeni, "Implementation of breadth-first search parallel to predict drug-target interaction in plant-disease graph," in *International Conference on Computer Science and Its Application in Agriculture (ICOSICA)*, pp. 1-5, 2020.

[7] B. Imbernon, A. Serrano, A. Bueno-Crespo, J. L. Abellan, H. Perez-Sanchez, and J. M. Cecilia, "METADOCK 2: a high-throughput parallel metaheuristic scheme for molecular docking," *Bioinformatics,* vol. 37, no. 11, pp. 1515-1520, 2021.

[8] I. Sanchez-Linares, H. Perez-Sanchez, J. M. Cecilia, and J. M. Garcia, "High-throughput parallel blink virtual screening using BINDSURF," *BMC Bioinformatics*, vol. 13, no. 14, pp. 1-14, 2012.

[9] T. I. Oprea, C. G. Bologa, B. S. Edwards, E. R. Prossnitz, and L. A. Sklar, "Post-high-throughput screening analysis: An empirical compound prioritization scheme," *Journal of Biomolecular Screening*, vol. 10, no. 5, pp. 419–426, 2005.

[10] G. M. Rishton, "Reactive compounds and in vitro false positives in HTS," *Drug Discovery Today*, vol. 2, no. 9, pp. 382–384, 1997.

[11] G. M. Rishton, "Nonleadlikeness and leadlikeness in biochemical screening," *Drug Discovery Today*, vol. 8, no. 2, pp. 86–96, 2003.

[12] S. L. McGovern, E. Caselli, N. Grigorieff, and B. K. Shoichet, "A common mechanism underlying promiscuous inhibitors from virtual and high-throughput screening," *Journal of Medicinal Chemistry*, vol 45, no. 8, pp. 1712–1722, 2002.

[13] V. P. Kamboj, "Herbal medicine," *Current Science*,

vol. 78, no. 1, pp. 35-39, 2000.

[14] S. K. Pal and Y. Shukla, "Herbal medicine: current status and the future," *Asian Pacific Journal of Cancer Prevention*, vol. 4, no. 4, pp. 281-288, 2003.

[15] P. Basnet and N. Skalko-Basnet "Curcumin: An anti-inflammatory molecule from a curry spice on the path to cancer treatment," *Molecules*, vol. 16, no. 6, pp. 4567–4598, 2011.

[16] D. Luebke, M. Harris, N. Govindaraju, A. Lefohn, M. Houston, J. Owens, et al., "GPGPU: general-purpose computation on graphics hardware," in *ACM/IEEE Conference on Supercomputing*, 2006.

[17] F. Li, Y. Ye, Z. Tian, and X. Zhang, "CPU versus GPU: which can perform matrix computation faster—performance comparison for basic linear algebra subprograms," *Neural Computing and Applications*, vol. 31, no. 8, pp. 4353–4365, 2019.

[18] NVIDIA Corporation, "CUDA Toolkit Documentation - v11.4.0," NVIDIA Corporation, Santa Clara, United States, 2021.

[19] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, "ZINC: a free tool to discover chemistry for biology," *Journal of Chemical Information and Modeling*, vol. 52, no. 7, pp. 1757–1768, 2012.

[20] F. Corpet, "Multiple sequence alignment with hierarchical clustering," *Nucleic Acids Research*, vol. 16, no. 22, pp. 10881–10890, 1984.

[21] D. Bajusz, A. Racz, and K. Heberger, "Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations?," *Journal of Cheminformatics*, vol. 7, no. 1, pp. 1-13, 2015.

[22] Y. C. Cao, Q. X. Deng, and S. X. Dai, "Remdesivir for severe acute respiratory syndrome coronavirus 2 causing COVID-19: An evaluation of the evidence," *Travel Medicine and Infectious Disease*, vol. 35, pp. 101647, 2020.

[23] Y. Furuta, T. Komeno, and T. Nakamuba, "Favipiravir (T-705), a broad spectrum inhibitor of viral RNA polymerase," in *Proceedings of the Japan Academy, Series B*, vol. 93, no. 7, pp. 449–463, 2017.

[24] R. S. Cvetkovic and K. L. Goa, "Lopinavir/Ritonavir", *Drugs*, vol. 63, no. 8, pp. 769–802, 2003.

[25] Z. N. Lei, Z. X. Wu, S. Dong, D-H. Yang, L. Zhang, Z. Ke, C. Zou, and Z-S. Chen, "Chloroquine and hydroxychloroquine in the treatment of malaria and repurposing in treating COVID-19," *Pharmacology & Therapeutics*, vol. 216, no. 1, pp. 107672, 2020.

[26] D. B. Mahmoud, Z. Shitu, and A. Mostafa, "Drug repurposing of nitazoxanide: can it be an effective therapy for COVID-19?," *Journal of Genetic Engineering and Biotechnology*, vol. 18, no. 35, pp. 1–10, 2020.

[27] A. Tan, L. Duan, Y. Ma, Q. Huang, K. Mao, W. Xiao, et al., "Is oseltamivir suitable for fighting against COVID-19: In silico assessment, in vitro and retrospective study," *Bioorganic Chemistry*, vol. 104, pp. 104257, 2020.

[28] T. F. de Koning-Ward, A. P. Waters, and S. Brendan, "Puromycin-N-acetyltransferase as a selectable marker for use in *Plasmodium falciparum*," *Molecular and Biochemical Parasitology*, vol. 117, no. 2, pp. 155-160, 2001.

[29] G. Lang, M. I. Mitova, G. Ellis, S. van der Sar, R. K. Phipps, J. W. Blunt, et al., "Bioactivity profiling using HPLC/microtiter-plate analysis: application to a New Zealand marine alga-derived fungus, *Gliocladium sp.*," *Journal of Natural Products*, vol. 69, no. 4, pp. 621-624, 2006.

[30] M. Isaka, S. Palasarn, S. Lapanun, and K. Sriklung, "Paecilodepsipeptide A, an antimalarial and antitumor cyclohexadepsipeptide from the insect pathogenic fungus *Paecilomyces cinnamomeus* BCC 9616," *Journal of Natural Products*, vol. 70, no. 4, pp. 675-678, 2007.

[31] S. R. Giacomelli, F. C. Missau, M. A. Mostardeiro, U. F. da Silva, I. I. Dalcol, N. Zanatta, and A. Morel, "Cyclopeptides from the bark of *Disaria americana*," *Journal of Natural Products*, vol. 64, no. 7, pp. 997-999, 2001

[32] A. G. Tolkushin, E. A. Luchinin, M. E. Kholovnya-Voloskova, and A. A. Zavyalov, "History of aminoquinoline preparations: from cinchona bark to chloroquine and hydroxychloroquinon," *Problemy Sotsial'noi Gigieny, Zdravookhraneniia i Istorii Meditsiny*, vol. 28 [special issue], pp. 1118-1122, 2020.

[33] R. Goto, K. Okura, H. Sakazaki, T. Sugawara, S. Matsuoka, and M. Inoue, "Synthesis and biological evaluation of triazole analogues of antillatoxin," *Tetrahedron*, vol. 67, no. 35, pp. 6659-6672, 2011.

[34] G. X. Zhou, E. K. Wijeratne, D. Bigelow, L. S. Pierson, H. D. VanEtten, and A. L. Gunatilaka, "Aspochalasins I, J, and K: three new cytotoxic cytochalasans of *Aspergillus flavipes* from the rhizosphere of *Ericameria laricifolia* of the Sonoran Desert," *Journal of Natural Products*, vol. 67, no. 3, pp. 328-332, 2004.

[35] Y. C. Shen, Y. B. Cheng, J. Kobayashi, T. Kubota, Y. Takahashi, Y. Mikami, et al., "Nitrogen-containing verticillene diterpenoids from the taiwanese soft coral *Cespitularia taeniata*," *Journal of Natural Products*, vol. 70, no. 12, pp. 1961-1965, 2007.