Aho-Corasick and Rabin-Karp Parallel With CUDA To Find Nucleotide Sequences In Genomic Sequences

\*Note: Sub-titles are not captured in Xplore and should not be used

Raden Jiwa Bumi Prajasantana   
School of Information Science and Technology  
Universitas Pelita HarapanTangerang, Indonesia  
01082220020@student.uph.edu

Callysa Tanjaya  
School of Information Science and Technology  
Universitas Pelita HarapanTangerang, Indonesia  
010822200xx@student.uph.eduJovan Christian   
School of Information Science and Technology  
Universitas Pelita HarapanTangerang, Indonesia  
010822200xx@student.uph.edu

Rifandi  
School of Information Science and Technology  
Universitas Pelita HarapanTangerang, Indonesia  
010822200xx@student.uph.eduHans Adriel  
School of Information Science and Technology  
Universitas Pelita HarapanTangerang, Indonesia  
010822200xx@student.uph.edu

Student  
School of Information Science and Technology  
Universitas Pelita HarapanTangerang, Indonesia  
010822200xx@student.uph.edu

***Abstract*—This electronic document is a “live” template and already defines the components of your paper [title, text, heads, etc.] in its style sheet. *\*CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract*. (***Abstract***)**

***Keywords—component, formatting, style, styling, insert (****key words****)***

# Introduction

This template, modified in MS Word 2007 and saved as a “Word 97-2003 Document” for the PC, provides authors with most of the formatting specifications needed for preparing electronic versions of their papers. All standard paper components have been specified for three reasons: (1) ease of use when formatting individual papers, (2) automatic compliance to electronic requirements that facilitate the concurrent or later production of electronic products, and (3) conformity of style throughout a conference proceedings. Margins, column widths, line spacing, and type styles are built-in; examples of the type styles are provided throughout this document and are identified in italic type, within parentheses, following the example. Some components, such as multi-leveled equations, graphics, and tables are not prescribed, although the various table text styles are provided. The formatter will need to create these components, incorporating the applicable criteria that follow.

Genomic sequences contain crucial data about living organisms, in our case, ours contain the data of the animal *house mouse* (*Mus musculus*). Genomic data are very vast and complex, containing hundreds of thousands of lines and can reach up to several gigabytes in text file alone.

Traditional string matching algorithms such as Rabin-Karp and Aho-Corasick are able to to perform on such datasets due to their nature. However, the more large and diverse datasets get, the more performance will degrade.

For decades, computer scientists have searched into either new methods or ways to improve current algorithms and one solution is to make them parallel and use GPU rather than entirely on CPU.

GPUs are huge devices compared to CPUs and were originally used solely to process graphical data. However today with the existence of CUDA and ways to do machine learning and data science on their GPUs, they are confident that it is 215x faster and improves will productivity. Indeed it is common knowledge in 2025 that most work and projects relating to data science and machine learning will always use GPU first and foremost.

The only downside at this moment, is that CUDA is for only NVIDIA GPUs and even then, only for the newer GPU models. Any other older NVIDIA GPUs or non-NVIDIA GPU models will require alternative way besides CUDA, however this paper will specifically focus on CUDA only.

Rabin-Karp is…. and with CUDA it will… BLABLABLABLABLABLA

Aho-Corasick meanwhile…. and when implemented with CUDA… BLABLABLABLABLA

Genomic sequences menyimpan data penting mengenai organisme hidup, dalam kasus ini, data yang digunakan berasal dari hewan house mouse (*Mus musculus*). Data genomik memiliki ukuran yang sangat besar dan kompleks, terdiri dari ratusan ribu baris dan bahkan bisa mencapai beberapa gigabyte hanya dalam format teks.

Algoritma string matching tradisional seperti Rabin-Karp dan Aho-Corasick memang mampu digunakan untuk memproses dataset semacam ini karena karakteristik algoritmanya. Namun, seiring dengan bertambah besarnya dan semakin beragamnya dataset, performa dari algoritma tersebut akan semakin menurun.

Selama beberapa dekade terakhir, para ilmuwan komputer telah berupaya mencari metode baru atau meningkatkan performa dari algoritma yang ada, dan salah satu solusinya adalah dengan membuat algoritma tersebut berjalan secara paralel menggunakan GPU, bukan hanya mengandalkan CPU.

GPU awalnya dirancang untuk memproses data grafis, dan ukurannya jauh lebih besar dibandingkan CPU. Namun, dengan munculnya CUDA dan berbagai metode untuk menjalankan machine learning serta data science menggunakan GPU, kini telah terbukti bahwa GPU dapat meningkatkan kecepatan eksekusi hingga 215 kali lipat dan secara signifikan meningkatkan produktivitas. Pada tahun 2025, sudah menjadi hal yang umum bahwa sebagian besar pekerjaan dan proyek terkait data science maupun machine learning selalu mengutamakan penggunaan GPU terlebih dahulu.

Satu-satunya kekurangan saat ini adalah bahwa CUDA hanya dapat digunakan pada GPU buatan NVIDIA, dan bahkan terbatas pada model-model terbaru. GPU NVIDIA versi lama maupun GPU dari vendor lain memerlukan pendekatan alternatif selain CUDA. Namun, penelitian ini akan secara khusus berfokus pada implementasi berbasis CUDA.

Rabin-Karp is…. and with CUDA it will… BLABLABLABLABLABLA

Aho-Corasick meanwhile…. and when implemented with CUDA… BLABLABLABLABLA

**1.1 Background and Motivation**

*String matching algorithm* merupakan suatu algoritma yang digunakan untuk mencari dan mencocokkan pola dari sebuah teks terhadap teks lain. Adanya algoritma ini mempermudah proses pencarian data spesifik dalam sebuah sistem *database* yang kompleks. *String matching algorithm* umum diaplikasikan dalam berbagai bidang di kehidupan sehari-hari, seperti dalam bidang bioinformatika, penerjemahan teks, pengecekan plagiarisme, dan lain-lain. Seiring dengan berkembangnya zaman, ukuran dari data semakin meningkat secara pesat sehingga penggunaan algoritma sekuensial kerapkali menghabiskan banyak waktu dan biaya komputasi yang tinggi. Oleh karena itu, komputasi paralel sering digunakan untuk mengatasi permasalahan ini untuk meningkatkan performa dan mengurangi waktu eksekusi.

Dari berbagai *string matching algorithm* yang ada, algoritma *Aho-Corasick* dan algoritma *Rabin Karp* merupakan contoh *string matching algorithm* yang dapat diparalelisasikan. Algoritma *Aho-Corasick* merupakan algoritma yang dikembangkan oleh Alfred V. Aho dan Margaret J. Corasick yang digunakan untuk mencari *multi-pattern* matching untuk string berukuran besar. Penelitian sebelumnya yang dilakukan oleh Thambawita et al. pada 2016 menunjukkan bahwa algoritma *Paralel Failur-Less Aho-Corasick* yang dioptimasi menunjukkan performa yang baik untuk dataset *DNA Sequence* karena dapat mengoptimasi penggunaan memory *cache* dari *Graphical Processing Unit* (GPU). Sementara itu, algoritma Rabin Karp merupakan algoritma pencocokan string yang menggunakan fungsi hash sebagai pembanding antara string yang dicari (m) dengan substring pada teks (n). Penelitian terdahulu yang dilakukan oleh Shah et al. pada 2018 menunjukkan bahwa versi paralel *Compute Unified Device Architecture* CUDA dari algoritma Rabin Karp dapat memiliki *speedup* besar. Selain itu, salah satu anggota untuk projek ini, Raden Jiwa Bumi Prajasantana, sudah melakukan penelitian Aho-Corasick juga sebelumnya berjudul “Performance Analysis of Non Linear Algorithm using GPU and CPU”, meskipun dapat pengetahuan dan pengalaman banyak, ini masih tidak menggunakan CUDA.

Penelitian ini bertujuan untuk menilai performa dari *string matching algorithm* paralel yang dieksekusi dengan GPU yang kompatibel dengan teknologi CUDA. Implementasi versi paralel dari masing-masing algoritma *Aho-Corasick* dan *Rabin Karp* akan diukur waktu eksekusi beserta konsumsi memori yang diperlukan. Selain itu, eksekusi versi serial dari kedua algoritma tersebut juga akan dilakukan untuk mengetahui kelebihan dengan komputasi paralel. Analisis komparatif dari kedua algoritma dilakukan untuk melihat masing-masing kelebihan dan kelemahan dari kedua algoritma tersebut.

**1.2. Problem Statement**

String matching adalah salah satu operasi yang dasar dalam ilmu komputer yang dimiliki dalam berbagai aplikasi, mulai dari pemrosesan teks sampai bioinformatika. Dalam analisis DNA, algoritma string matching ini digunakan untuk mencari urutan nukleotida dalam sekuens genomik yang sangat panjang. Tantangan utama yang melakukan pencocokan string pada skala besar adalah waktu eksekusi yang tinggi dan konsumsi sumberdaya komputasi yang besar, terutam a ketika menggunakan metode berbasis CPU.

Algoritma String matching berbasis CPU ini memiliki keterbatasan dalam efisiensi karena bersifat sekuensial. Ketika melakukan analisis dengan dataset yang ukuran cukup besar, seperti dalam genom manusia yang terdiri dari banyak sekali nukleotida, algoritma matching konvensional akan mengalami penurunan performa yang cukup signifikan. Hal ini buat melakukan analisis menjadi lambat dan tidak praktis untuk aplikasi di dunia nyata seperti deteksi mutasi genetik, pengolahan data biologis jumlah besar, dan identifikasi penyakit. Oleh karena itu, pendekatan yang berbasis komputasi paralel menggunakan GPU untuk menjadi solusi potensial untuk meningkatkan kecepatan dan efisiensi dalam pencocokan string dalam bioinformatika.

Dalam melakukan penelitian ini, kami fokus dalam implementasi dan optimasi dua algoritma string matching yaitu Rabin-Karp dan Aho-Corasick, menggunakan CUDA (Compute Unified Device Architecture) untuk menjalankan proses secara paralel di GPU. Rabin-Karp ini menggunakan teknik hashing untuk mencocokan pola dalam teks dengan cepat, sedangkan Aho-Corasick menggunakan automation untuk melakukan pencocokan multi-pattern yang lebih efisien. Meskipun kedua algoritma ini biasanya digunakan dalam aplikasi, tapi belum banyak penelitian secara langsung membandingkan performanya dalam lingkungan paralel berbasis CUDA, khususnya dalam konteks analisa DNA.

**1.3. Research Objectives**

Tujuan dari penelitian ini adalah sebagai berikut:

1. Menerapkan dan menganalisis kinerja algoritma Aho-Corasick dan Rabin-Karp dalam komputasi paralel menggunakan CUDA.
2. Menggunakan kumpulan data Genome Assembly GRCm39 untuk mencari urutan nukleotida dalam urutan genomik yang panjang.
3. Memperoleh hasil output dari proses pencocokan string paralel.
4. Membuat grafik dan visualisasi output
5. Membandingkan hasil kedua algoritma dalam hal waktu eksekusi dan efisiensi, workload GPU…
6. Menganalisis dan menginterpretasikan perbedaan kinerja antara kedua algoritma.

**1.4. Contributions and Paper Organization**

Adapun kontribusi yang hendak dicapai dalam penelitian ini adalah sebagai berikut:

1. **Improved Parallel Rabin-Karp Algorithm Using Compute Unified Device Architecture**: Paper ini berhubungan dengan penelitian kami karena paper ini membahas implementasi paralel Rabin Karp dengan menggunakan CUDA, yang sesuai dengan kebutuhan dalam proyek kami. Kemudian, implementasi yang digunakan akan lebih baik dibandingkan versi yang reguler. Namun, penelitian ini diuji pada komputer yang lama, sehingga paper kami dapat berkontribusi dengan membuktikan relevansi penelitian tersebut, terutama ketika melakukan pengujian dalam komputer yang modern.
2. **An Improved Hashing Approach for Biological Sequence to SolveExact Pattern Matching Problems**: Penelitian ini memiliki hubungannya dalam project kami karena paper ini menjelaskan secara detail tentang logika dibalik perencanaan dan penggunaan algoritma mereka. Selain itu, penelitian ini semakin mirip dengan project yang kami buat karena sama-sama menggunakan dataset DNA. Tapi, penelitian ini tidak menggunakan CUDA, jadi paper ini kami akan menjelaskan sejauh mana perbedaan performa ketika menggunakan CUDA.
3. **An Optimized Parallel Failure-less Aho-Corasick Algorithm for DNA Sequence Matching**: Alasan relevansinya mirip sama poin sebelumnya, tapi kali ini lebih fokus dalam menggunakan algoritma Aho-Corasick.
4. **Performance Analysis of Non Linear Algorithm using GPU and CPU**: Paper dari projek kelas Analisis Algoritma oleh Raden Jiwa Bumi Prajasantana, Jonathan Tiong, dan Dheana Laurens, ini sangat penting untuk projek ini, paper ini melakukan Aho-Corasick parallel tetapi tanpa CUDA atau GPU. Dengan paper ini tentang CUDA, mesti akan tahu perbedaannya Aho-Corasick reguler single thread, reguler multi thread, parallel single thread, parallel multi thread, dan parallel CUDA.

# Cuda and string matching

## CUDA Overview

*Compute Unified Device Architecture* (CUDA) merupakan platform komputasi paralel yang dikembabgkan oleh NVIDIA untuk memanfaatkan kemampuan pemrosesan dari *Graphical Processing Unit* (GPU). CUDA menyediakan model pemrograman dengan abstraksi yang sederhana untuk organisasi *thread* dan manajemen memori. Pemrograman CUDA juga mendukung manajemen memori hierarkis dan sinkronisasi antar *threads*, yang memberikan kontrol penuh terhadap arsitektur GPU. CUDA mendukung berbagai bahasa pemrograman seperti C, C++, dan Fortran.

Implementasi CUDA dalam algoritma *string matching* menunjukkan peningkatan performa yang signifikan untuk dataset dalam jumlah besar. Penelitian terdahulu yang telah dilakukan menunjukkan bahwa paralelisasi dari algoritma seperti Rabin Karp, Aho Corasick, dan Knuth Morris Pratt memiliki percepatan apabila dibandingkan dengan versi sekuensialnya,

## String Matching Algorithm dan Cuda

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

# Methodology

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections A-D below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you.

## *Abbreviations and Acronyms*

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

## *Units*

* Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) English units may be used as secondary units (in parentheses). An exception would be the use of English units as identifiers in trade, such as “3.5-inch disk drive”.
* Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.
* Do not mix complete spellings and abbreviations of units: “Wb/m2” or “webers per square meter”, not “webers/m2”. Spell out units when they appear in text: “. . . a few henries”, not “. . . a few H”.

Identify applicable funding agency here. If none, delete this text box.

* Use a zero before decimal points: “0.25”, not “.25”. Use “cm3”, not “cc”. (*bullet list*)

## *Equations*

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

Number equations consecutively. Equation numbers, within parentheses, are to position flush right, as in (1), using a right tab stop. To make your equations more compact, you may use the solidus ( / ), the exp function, or appropriate exponents. Italicize Roman symbols for quantities and variables, but not Greek symbols. Use a long dash rather than a hyphen for a minus sign. Punctuate equations with commas or periods when they are part of a sentence, as in:

*a**b* 

Note that the equation is centered using a center tab stop. Be sure that the symbols in your equation have been defined before or immediately following the equation. Use “(1)”, not “Eq. (1)” or “equation (1)”, except at the beginning of a sentence: “Equation (1) is . . .”

## *Some Common Mistakes*

* The word “data” is plural, not singular.
* The subscript for the permeability of vacuum *μ*0, and other common scientific constants, is zero with subscript formatting, not a lowercase letter “o”.
* In American English, commas, semicolons, periods, question and exclamation marks are located within quotation marks only when a complete thought or name is cited, such as a title or full quotation. When quotation marks are used, instead of a bold or italic typeface, to highlight a word or phrase, punctuation should appear outside of the quotation marks. A parenthetical phrase or statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.)
* A graph within a graph is an “inset”, not an “insert”. The word alternatively is preferred to the word “alternately” (unless you really mean something that alternates).
* Do not use the word “essentially” to mean “approximately” or “effectively”.
* In your paper title, if the words “that uses” can accurately replace the word “using”, capitalize the “u”; if not, keep using lower-cased.
* Be aware of the different meanings of the homophones “affect” and “effect”, “complement” and “compliment”, “discreet” and “discrete”, “principal” and “principle”.
* Do not confuse “imply” and “infer”.
* The prefix “non” is not a word; it should be joined to the word it modifies, usually without a hyphen.
* There is no period after the “et” in the Latin abbreviation “et al.”.
* The abbreviation “i.e.” means “that is”, and the abbreviation “e.g.” means “for example”.

An excellent style manual for science writers is [7].

# Result and Analysis

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

## *Authors and Affiliations*

**The template is designed for, but not limited to, six authors.** A minimum of one author is required for all conference articles. Author names should be listed starting from left to right and then moving down to the next line. This is the author sequence that will be used in future citations and by indexing services. Names should not be listed in columns nor group by affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization).

### *For papers with more than six authors:* Add author names horizontally, moving to a third row if needed for more than 8 authors.

### *For papers with less than six authors:* To change the default, adjust the template as follows.

#### *Selection:* Highlight all author and affiliation lines.

#### *Change number of columns:* Select the Columns icon from the MS Word Standard toolbar and then select the correct number of columns from the selection palette.

#### *Deletion:* Delete the author and affiliation lines for the extra authors.

## *Identify the Headings*

Headings, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

Component heads identify the different components of your paper and are not topically subordinate to each other. Examples include Acknowledgments and References and, for these, the correct style to use is “Heading 5”. Use “figure caption” for your Figure captions, and “table head” for your table title. Run-in heads, such as “Abstract”, will require you to apply a style (in this case, italic) in addition to the style provided by the drop down menu to differentiate the head from the text.

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced. Styles named “Heading 1”, “Heading 2”, “Heading 3”, and “Heading 4” are prescribed.

## *Figures and Tables*

#### *Positioning Figures and Tables:* Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation “Fig. 1”, even at the beginning of a sentence.

1. Table Type Styles

| **Table Head** | **Table Column Head** | | |
| --- | --- | --- | --- |
| ***Table column subhead*** | ***Subhead*** | ***Subhead*** |
| copy | More table copya |  |  |

1. Sample of a Table footnote. (*Table footnote*)
2. Example of a figure caption. (*figure caption*)

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity “Magnetization”, or “Magnetization, M”, not just “M”. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write “Magnetization (A/m)” or “Magnetization {A[m(1)]}”, not just “A/m”. Do not label axes with a ratio of quantities and units. For example, write “Temperature (K)”, not “Temperature/K”.

# Conclusion

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

##### References

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**

