IUPACpalV2: Efficient Inverted Repeat Identification via Web Tool

Kawchar Husain (2018331085)

Samia Preity (2018331008)

Abstract

An inverted repeat (IR) is a single stranded sequence of nucleotides with a subsequent downstream sequence consisting of its reverse complement. Any sequence of nucleotides appearing between the initial component and its reverse complement is referred to as the gap (or the spacer) of the IR. The gap's size may be of any length, including zero. We developed a website to find inverted repeats with a convenient user interface, a library to store results, an admin system for overall supervision, and of course optimization of the algorithm.

Motivation

Studying inverted repeats and palindromes has become very important from a biological perspective. For instance, inverted repeats and palindromes act as an indicator of the probability of cancer existence. Various algorithmic tools and software have been published to enable the study of IRs in genomes including EMBOSS and IUPACpal. EMBOSS software can meaningfully process IUPAC encoded sequences properly, but IUPACpal works with better efficiency and identifies many previously unidentified IRs when compared with EMBOSS. But IUPACpal is a Command Line Interface System where its users are biologists who prefer a handy user interface instead of a black and white shell window. So, the need for a website with an improved UI arises.

Objective

Making a user-friendly website to efficiently identify the Inverted Repeats (IR) in “The International Union of Pure and Applied Chemistry (IUPAC)” encoded DNA sequence.

Features

I. Efficiently finding inverted repeats in IUPAC-encoded DNA.
II. User-friendly interface.
III. Besides normal IR finding gapped inverted repeat with mismatches.
IV. Storing results of specific sequences in a database.
V. Providing an admin panel to supervise the system.
VI. Search/update results in the database.
VII. Optimizing existing algorithm.
VIII. Showing statistics.

Technologies

NextJS, TypescriptJS, C++, NodeJS, ExpressJS, MySQL, Firebase, MantineUI, CSS, HTML.

Methodology

The web project utilizes a combination of various technologies to provide a user-friendly interface for identifying inverted repeats in IUPAC-encoded DNA sequences. The methodology involved in building this web project can be divided into several parts.

Frontend:

NextJS is used as the primary framework for the frontend. It provides server-side rendering, which makes the website more SEO-friendly and provides a better user experience. TypescriptJS is used for type checking, making the codebase more robust and easier to maintain. MantineUI is used as a UI library, providing pre-built components and styles for designing the user interface. CSS and HTML are used for styling and markup.

Backend:

NodeJS and ExpressJS are used to create a lightweight and efficient server for the web project. The server communicates with the frontend to provide data and handle requests. C++ is used as the primary language for the algorithm to identify inverted repeats in IUPAC-encoded DNA sequences. The C++ code is compiled as a shared object, which can be loaded into the NodeJS server and executed

Database:

We are using MySQL as a relational database management system to store input parameters and input/output file links. Firebase, a cloud-based platform, is used to store large input and output files. By using both, we can efficiently manage and store data, ensuring our system can handle large amounts of data and perform well.

Algorithm and data structures used:

- Manber-Myers algorithm to construct suffix arrays
- Kasai algorithm to compute the LCP array
- Range minimum query (RMQ) data structure to efficiently compute the LCP range
- Kangaroo method to efficiently identify mismatched pairs
- Dynamic programming to extend inverted repeats
- Palindromic Tree to efficiently check palindrome

Implementation of finding IR:

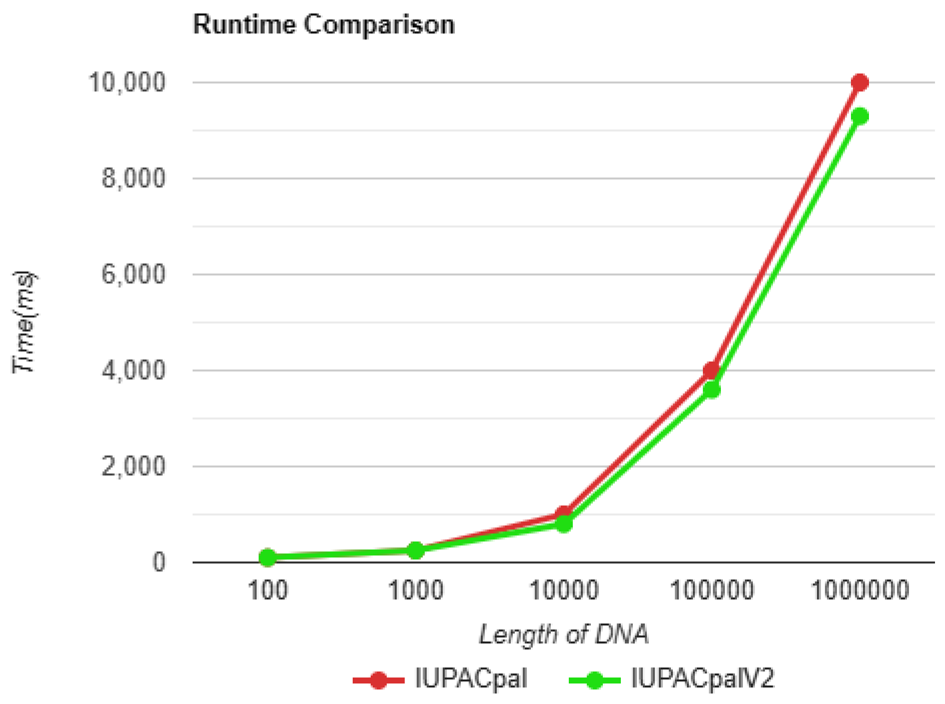
To efficiently identify IRs in IUPAC-encoded DNA sequences, we implemented an algorithm involving three main steps:

1. Preprocessing for Kangaroo Method:
    - Create a complement copy of the input DNA sequence.
    - Construct suffix arrays for the original and complement sequences using the Manber-Myers algorithm.
    - Compute the LCP array for both suffix arrays using Kasai's algorithm.
2. Identification of Maximal Palindromes:
    - Traverse the LCP array for the original sequence.
    - For each element in the LCP array, check if the corresponding suffixes in the original sequence form a palindrome.
    - To check if a suffix is a palindrome, compare the characters at the start and end of the suffix, then move inwards toward the center until the entire suffix has been checked.
    - If the suffix is a palindrome, check if a prefix with the same length within the same LCP range exists.
    - If a prefix with the same length within the same LCP range exists, then the substring between the prefix and suffix is a maximal palindrome.
3. Identification of Inverted Repeats:
    - Iterate over every position in the input sequence.
    - For each position, use the Kangaroo method to identify all mismatched pairs equidistant from the current position.
    - Calculate only the mismatch locations needed for a given set of parameters to maintain efficiency.
    - For each identified mismatched pair, consider a range of gap sizes and extend the inverted repeat to both the left and right, as long as the number of mismatches stays below some maximum value.

To optimize the algorithm, we have improved the complexity of the existing IUPACpal algorithm by a constant factor. In the original implementation of IUPACpal, the complexity for the range minimum query was O(logN), but in our IUPACpalV2 implementation, we have reduced it to O(1). The overall time complexity of our algorithm is O(N log N), where N is the length of the input sequence.

Run-time analysis

For small DNA lengths, the runtime difference between IUPACpal and IUPACpalV2 is negligible. However, for larger DNA sequences, IUPACpalV2 is slightly faster, with a runtime difference of approximately 1-2%.



Past Works

- IUPACpal: efficient identification of inverted repeats in IUPAC-encoded DNA sequences
- Generic Repeat Finder: A High-Sensitivity Tool for Genome-Wide De Novo Repeat Detection
- Long Inverted Repeats Are an At-Risk Motif for Recombination in Mammalian Cells
- detectIR: A Novel Program for Detecting Perfect and Imperfect Inverted Repeats Using Complex Numbers and Vector Calculation
- PCIR: a database of Plant Chloroplast Inverted Repeats
- Inverted Repeat Structure of the Human Genome: The X-Chromosome Contains a Preponderance of Large, Highly Homologous Inverted Repeats That Contain Testes Genes
- airpg: automatically accessing the inverted repeats of archived plastid genomes
- RepeatsPlus — program for finding motifs and repeats in data sequences
- MITE Digger, an efficient and accurate algorithm for genome wide discovery of miniature inverted repeat transposable elements
- The inverted repeat as a recognizable structural feature in supercoiled DNA molecules
- Chloroplast DNA rearrangements are more frequent when a large inverted repeat sequence is lost
- Lost and Found: Return of the Inverted Repeat in the Legume Clade Defined by Its Absence
- Miniature inverted-repeat transposable elements: discovery, distribution, and activity
- Instability of long inverted repeats within mouse transgenes
- Replication stalling at unstable inverted repeats: Interplay between DNA hairpins and fork stabilizing proteins