InstaCircos: a Web Application for Fast and Interactive Circular Visualization of Large Genomic Data (Work in Progress)

Gaia Ghidoni, Riccardo Martoglia
FIM - University of Modena and Reggio Emilia, Italy
gaia.ghid@gmail.com, riccardo.martoglia@unimore.it

Cristian Taccioli, Chiara Vischioni
MAPS - University of Padova, Italy
cristian.taccioli@unipd.it, chiara.vischioni@phd.unipd.it

***Abstract*—One of the most effective visualizations for genomics data is the circular one, supported by popular packages and visualization suites. Many tools are available, however most of them share a number of negative points including limited ease of installation/usage, slow performance and memory limitations (making them unfeasible for very large genomes such as the human one) and non interactivity. In this paper we present the ongoing work on *InstaCircos*, a web application born from the scientific collaboration between Big Data Analytics and Bioinformatics researchers and aiming at overcoming the available tools' limitations. It provides advanced visualization features through an easy to use web interface and offers *interactive* functionalities and near *real-time* performances thanks to an integrated big data management back-end based on MongoDB.**

***Index Terms*—Genomic data visualization, interactive visualization, circular layout, big data management, web application.**

I. Introduction

When dealing with scientific data, it is essential for researchers to have tools allowing them to efficiently view huge quantities of data with effective visualizations. This is particularly true for bioinformatics research, where the size of the genomics data coupled with the fact that the analyses often depend on several still unexplored and unclear factors (think for instance about cancer research), makes the use of novel data analysis and, in particular, advanced and innovative data visualization techniques an essential matter.

When working with genomics data, one of the most effective visualizations is the circular one, supported by popular packages and visualization suites. Circos [1], [2], written in Perl, is the predominant package used for genome feature visualization. Other available options are RCircos [3], [4], omicCircos [5], [6], Circoletto [7], [8] and PyCircos [9].

The many options offered in the panorama of currently available tools shows that (circular) genomic visualization is an hot topic in current research. However, available tools share a number of negative points from multiple points of views:

- *Limited ease of use*: the installation and configuration of these tools requires technical knowledge on command-line and Perl (or Python) scripts, which poses a certain degree of difficulty for users lacking coding experience;
- *Slow performance and memory limitations*: while most tools can produce a visualization of small genomes in few minutes, the representation of medium-size genomes is typically a very long operation (hours). Moreover, the visualization of very large genomes such as the human one is often not possible: this is due not only to slow performances but also to the fact that these tools typically require all data to be kept in main memory;
- *Non interactivity*: available tools provide, after a certain amount of processing time, a static image of the data, which are perfectly suitable for a publication, but are not able to support user interaction which is recognized as a key aspect in biological data exploration [10].

In this paper we present an ongoing work we are conducting on a novel tool for interactive circular visualization of genomic data, named *InstaCircos*. InstaCircos aims at overcoming the available tools' limitations, providing through an easy to use web interface advanced visualization features offering *interactive* functionalities and near *real-time* performances.

The contributions of InstaCircos, which will be made publicly available as a web application once finished, are the following:

- *Circular visualization:* InstaCircos shows genomic data (including chromosome, coding DNA sequence (CDS) density and link information) in a familiar circular layout, which is particularly advantageous for the immediate feedback and for the visual impact of the result;
- *No size restrictions:* by means of an integrated big data management back-end based on MongoDB the tool enables the visualization of very large genomes (including human ones) without imposing any memory requirement on the user;
- *Near-instant visualizations:* the visualizations are generated in real-time from the cached data, enabling data exploration through different forms of user interaction on the generated graphs;
- *User-friendliness:* the tool does not require any coding skill or installation.

As far as we know, there are no tools or web applications of this kind offering all the above features.

The paper is organized as follows: in Section II we discuss related works, Section III gives a high-level overview of the application and its architecture, Section IV is devoted to the description of the data management strategies and integrated database, Section V discusses the client UI and user interaction. Finally, Section VI analyzes the development stages and the obtained performances, while Section VII concludes the paper, discussing future works.

II. Related Works

Many tools have been recently proposed in the field of genomic data visualization. Circos [1] is one of the most used and well-known among researchers and one of the first in proposing a circular layout, ideal for exploring relationships between objects or positions. Circos has appeared in many publications [11], both scientific and general ones, and it has changed the way the scientific community visualizes genomic alterations. Circos is written in Perl and its installation and configuration require knowledge on command-line and Perl script, making it non ideal for non coding experts.

Alternatives to Circos have been developed to (partially) address the issue, providing easy installation but, on the other hand, still requiring setup and coding-skills from the users. Among these application, we recall R packages such us RCircos [3] and omicCircos [5], which, nevertheless, require users to be familiar with R programming language. Further options, always requiring coding skills, are Circoletto [7], a visualization suite written in Perl, and pyCircos, written in Python. Circoletto has been developed in the specific context of bringing together in an efficient implementation BLAST (Basic Local Alignment Search Tool) and Circos functionalities. pyCircos [9] is a Python-based package able to represent genome features from the input of a Genbank [12] format file.

All the above mentioned options do not offer effective data management solutions allowing smart data allocation; instead, they require the full data to be kept in main memory, making the storage for the display of very large genomes often unfeasible (if not even impossible). For instance, pyCircos works efficiently with small genomes, such as the one of Archea microorganisms, but when managing bigger files, including the human genome, it generates memory errors making it impossible to generate plots. Moreover, interactivity is typically not supported.

The only web application option we are aware of is ClicO FS [13], [14], a user-friendly web service, developed using R and Shiny. The goal of ClicO is mainly to automate standard Circos installation and usage, making it run directly from a web browser. However, the shortcomings of the standard Circos remain, including the production of a static image (the "interaction" is basically for choosing the parameters to generate the image) and the limitation to small genomes.

InstaCircos, on the other hand, enables the efficient visualization even of very large genomes, also offering interactivity features to the generated plots.

III. High-level Overview

InstaCircos is presented as a Web Application, making all the visualization/analysis functions instantly available and easily accessible to researchers. The application includes a server side part for data management and a client side for managing the visualization / interaction / UI (see Figure 1). InstaCircos is being developed in Python using Dash[1] – a Python framework written on top of Flask, Plotly and React.js – for the UI components. Server side data management exploits Biopython[2], the open-source collection of tools for computational biology, and MongoDB[3], the document-oriented database.

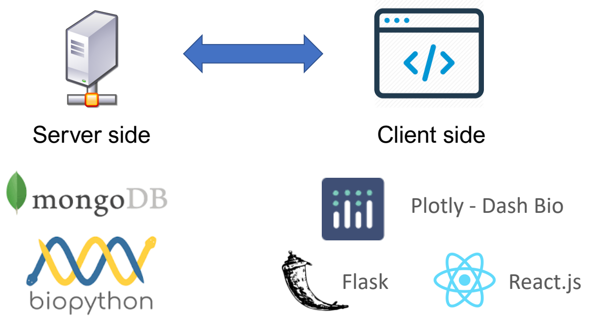


Fig. 1. InstaCircos architecture and technologies: server side and client side

InstaCircos allows users to select among a number of scientifically notable genomes, including very large ones such the human one. Such genomes are stored in the platform integrated database and ready to be visualized. Genome data (data from GenBank collection and additional elaborated data including link information from sequence alignment processing) is stored in the database to speed up the data retrieval at plot time, making it possible to create circular plots of large genomes without a huge parsing overhead. The use of a database also reduces the memory usage by retrieving only the data of interest for generating the plots. Genome file parsing, preprocessing, storing and indexing is optimized to be efficiently performed, therefore we plan to also support user-provided file storing and visualization in the final version of our tool. The format and details of the data and its processing, the schema of the database and how it is indexed and queried will be discussed in details in Section IV.

Once the genome of interest is selected, different kinds of features can be retrieved from the database through targeted queries and displayed in concentric tracks inside the chromosomes circular representation. After a minimal processing of the query result the data is ready to be passed to the the visualization component (based on Dash Bio) for the graphic rendering. The output can include up to 6 kinds of plot types, including heatmap, highlight, scatter, line, histogram and chords. Figure 2 shows an example plot representing the human genome, with the 24 labelled chromosomes in the outermost circle and additional data shown in the inner tracks (see caption for details). All generated plots support interactivity by means of mouse hover or mouse click events.

[1]http://plotly.com/dash/
[2]http://biopython.org/
[3]http://www.mongodb.com/

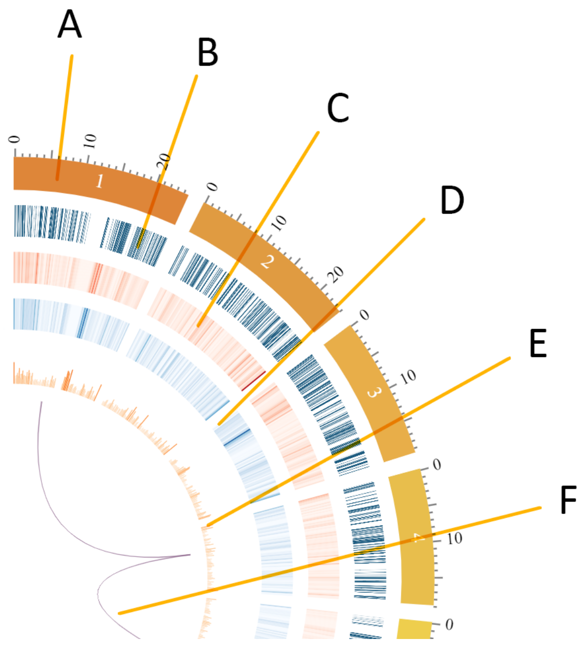


Fig. 2. Details on the human genome plot. The chromosomes, labeled with names, are arranged in the outer circle **(A)**, on which the length scale is displayed. The first highlight track **(B)** represents all CDS regions located in the forward DNA strand, and it's followed by two heatmap tracks that show the CDS density respectively on the forward **(C)** and on the reverse strand **(D)**. The fourth track reports the gene density on both strands **(E)**. Finally, the innermost tracks shows some sample links between chromosomes **(F)**.

More details on the visualization and interaction features are discussed in Section V.

IV. Data Management and Database

In this section we describe the details on the source and format of the data on which InstaCircos is based: from the raw genomic data, to its pre-processing, to the final database and its indexing and querying.

Genome files from the GenBank collection and their assembly report files are parsed with Biopython and stored in MongoDB. Additional data resulting by usage of LASTZ sequence alignment program[4] is also stored in the database and retrieved to create chords plots.

A. Raw data format and sources

The starting raw data come from the GenBank database: the NIH genomic sequence database, designed to provide access to the most up-to-date and comprehensive DNA information. In particular, the files of interest are *GBFF* (GenBank Flat File) files and *assembly report* files. The GBFF format is a way of representing nucleotide sequences including metadata annotation. It is presented as a list of *loci*: each locus represents a genome section, which may be a chromosome or other assembly levels such as scaffolds and contig (unplaced or unlocalized sequence) and it is completed with the genomic sequence and a list of genomic features. The assembly report file is a tab-delimited text file reporting name, role, sequence

[4]http://github.com/lastz/lastz/

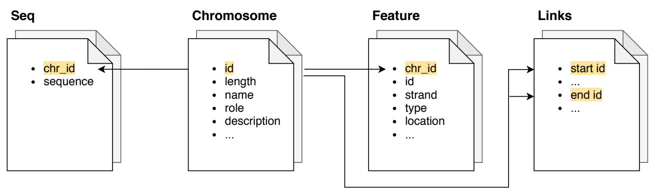


Fig. 3. Schema of the InstaCircos database in MongoDB

length and sequence accession version of each object in the assembly.

B. GBFF Data pre-processing

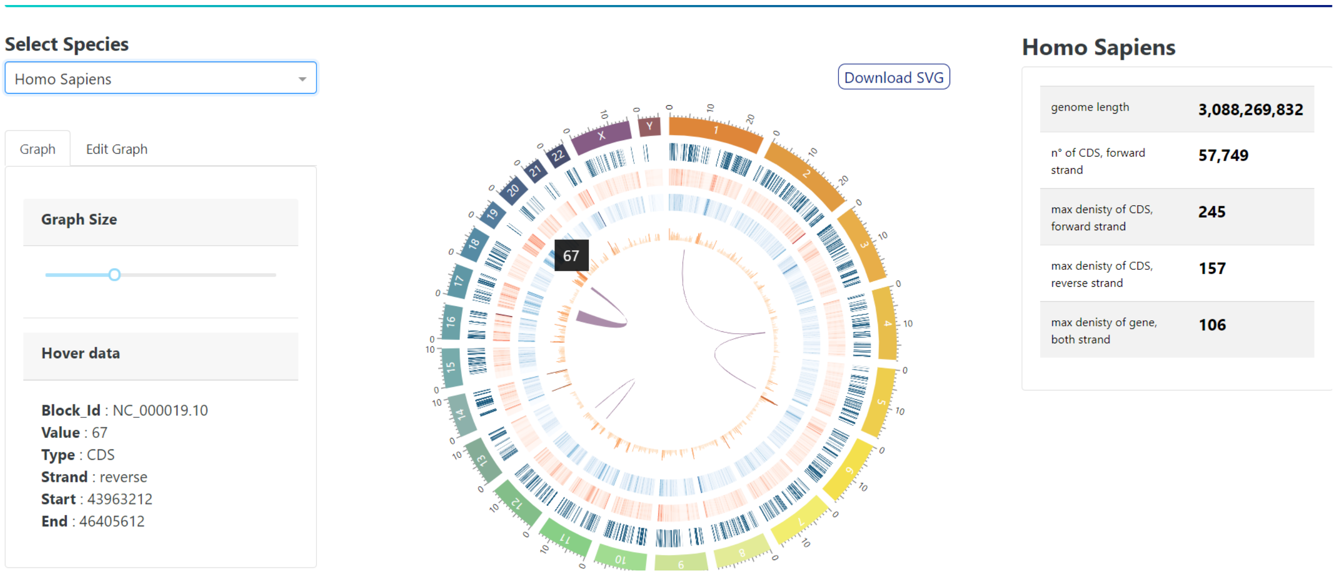
Once a genome to be stored in the database is selected, the assembly report file is read to retrieve the sequence role of all the sequences in the GBFF file. The sequences of interest are, in fact, the ones with the highest assembly level: the chromosomes. This information will be crucial for the retrieval at plot time and it will be stored in the database alongside the GBFF file's data. The GBFF file is read and parsed with the Biopython library, which provides an input-output interface for genomic files.The retrieved attributes are:

- **Id**: sequence identifier;
- **Annotations**: Python dictionary containing various information, link source and reference;
- **Description**: a string containing the sequence description;
- **Features**: includes the following attributes:
  - **Location**: feature location inside the genomic sequence;
  - **Type**: feature type (CDS, rRNA, exons, etc.);
  - **Strand**: DNA strand on which the feature is located. Value 1 indicates the forward strand, while value -1 indicates the reverse strand;
  - **Id**: feature identifier;
  - **Qualifiers**: Python dictionary with additional information;
- **Seq**: a string with the genomic sequence (a complete list of the nucleotides that make up the sequence) and its associated alphabet.

After a minimal rearrangement of the above information so to meet the document-like format required by MongoDB, the genome date is ready to be loaded in the database.

C. Sequence alignment analysis

In order to support link data visualization, besides the GBFF file preprocessing, the chromosome sequences are read and stored in FASTA format file (text-based format for representing genomic sequences, possibly preceded by the sequence name and comments) to be used as input data for the sequence alignment analysis phase, for which LASTZ is exploited. The links that will be extracted identify the reverse complements within the chromosomes, which InstaCircos will be able to graphically visualize. Since the output of the LASTZ command grows almost quadratically, we adopt specific filtering criteria in order to filter the results by excluding links of lesser importance and extracting only the most relevant ones



Select Species

Homo Sapiens

Graph | Edit Graph

Graph Size

Hover data

Block_Id : NC_000019.10
Value : 67
Type : CDS
Strand : reverse
Start : 43963212
End : 46405612

Homo Sapiens

genome length	3,088,269,832
n° of CDS, forward strand	57,749
max denisty of CDS, forward strand	245
max denisty of CDS, reverse strand	157
max denisty of gene, both strand	106

Fig. 4. InstaCircos web interface showing a sample plot of human genome: at the sides, a summary and a multi-tab panel add information and interactivity

(e.g., search in sequences longer than 100 bases, percentage of matching bases of at least 99%). This makes links extraction feasibile for all chromosomes (including the largest ones): for instance, for human chromosome 1 the output is reduced to 29MB (instead of over 900GB).

An example of LASTZ output file is a tab-delimited text file as the following:

name1	start1	end1	name2	start2+	end2+
chr22	31112257	31112301	chr22	50805867	50805911
chr22	18531481	18531516	chr22	50805868	50805903
chr22	25482021	25482052	chr22	50805668	50805699

Each row represents a link between two genomic regions (chromosome 22 in the example), both described with the starting and ending position in the target sequence.

D. InstaCircos Database

All the described data are then stored in a document-oriented database. The adoption of MongoDB is motivated by the scale-out architecture and the powerful and flexible data model that perfectly adapts to the genomic data structure, which, sometimes, may be incomplete.

The database is composed of four collections (Figure 3):

- **Chromosome**: this collection holds general information about each sequence;
- **Feature**: each document in the collection represents a feature. For each feature the following information are stored: the sequence identifier in which the feature is located; the actual location of the feature inside the sequence, written as a starting and ending coordinate; if present, a qualifiers sub-document, with additional data. If the feature is a set of non-contiguous sequences, an additional field stores the list of start-end coordinates;
- **Links**: this collection holds the alignment links, each stored as a document containing two sub-documents, for the beginning and the end of the link;
- **Seq**: this collection stores the genomic sequence itself, i.e., the complete list of the nucleotides, as a string.

Since genomic sequences can be long enough to exceed the database's document size limit, which is 16 megabytes, the GridFS API provided by MongoDB is exploited.

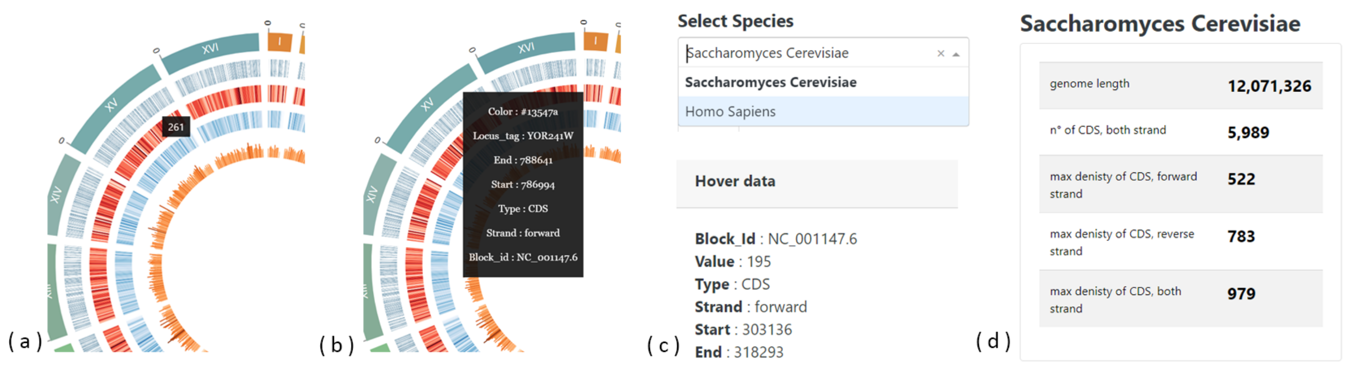
An instance of MongoDB database is kept for each stored genome, providing the possibility to select different kinds of sample species for generating the plots.

E. Dynamic data retrieval

With genome data pre-processed and stored in the database, data retrieval is performed at plot time, making it possible to create circular plots of large genomes by extracting only the needed information. More specifically, data is retrieved through the following queries w.r.t. the collections discussed in the previous section:

- the "chromosome" collection is queried to retrieve basic information about all the genomic sequences with a specific role and to retrieve data of single sequences, selected by its identifier;
- the "feature" collection's documents are retrieved based on the chromosome identifiers which they belong to and their feature type. If necessary, the DNA strand is also specified;
- queries on the "links" collection are used to retrieve all links starting in a specified sequence, or the links which occur between two sequences;
- finally, the "seq" collection holding the genomic sequences is queried through sequence identifiers.

An important note on link management: representing all the links retrieved with LASTZ and stored in the database would be of little use, as it would make the graph unreadable. Since the main interest regarding reverse complement sequences is to have information on the links that fall within genes, InstaCircos exploits an automatic *link filtering* phase, keeping



Color : #13547a
Locus_tag : YOR241W
End : 788641
Start : 786994
Type : CDS
Strand : forward
Block_id : NC_001147.6

Select Species

Saccharomyces Cerevisiae

Saccharomyces Cerevisiae

Homo Sapiens

Hover data

Block_Id : NC_001147.6
Value : 195
Type : CDS
Strand : forward
Start : 303136
End : 318293

Saccharomyces Cerevisiae

genome length	**12,071,326**
n° of CDS, both strand	**5,989**
max denisty of CDS, forward strand	**522**
max denisty of CDS, reverse strand	**783**
max denisty of CDS, both strand	**979**

Fig. 5. UI details: (a) density and (b) sequence information shown on mouse over; (c) species selection dropdown; (d) genome/tracks summary table.

only those that start and end within genes as candidates for representation. Finally, indexes on all collections support the efficient execution of all queries.

V. Client UI and Interaction

InstaCircos graphical interface is developed using Dash and Dash-bio framework. Dash is ideal for building data visualization apps with highly custom user interfaces. Moreover, since Dash apps are viewed in the web browser, InstaCircos interface is inherently cross-platform and mobile ready.

Figure 4 shows the web page UI of the current prototype of InstaCircos. See also Figure 5 for some details on the interface features and interactions. The focus of the user interface is on the graph visualization. The plot is shown in a central position, while at its sides a summary panel and a multi-tab panel are placed to add information and interactivity.

A dropdown, placed under the application logo, allows the user to choose a species from those available in the application's database. Once a species is selected, its genome is efficiently represented in the form of a circular plot (see Section VI for implementation and performance details).

The plot can be zoomed and moved inside its canvas, and details of the data represented in the graph are shown as soon as the mouse hover on a plot component, such us density value, feature name and position. A button on the top of the plot makes it possible to download a vector image (SVG image format) of the current shown graph.

The right-side panel is dynamically created on plot creation and update, reporting some summary information of each track, as well as the total length of the genome visualized. The summary data can be either the number of retrieved features (e.g., the number of genes in the forward strand), or the maximum value for a track reporting a feature density.

On the right-side panel, the first tab shows, on mouse hover on the plot, all the details available for the selected component, as well as a slider to interactively change the plot size.

As to link visualization, InstaCircos offers some functions that are not usually found on similar tools. It exploits link filtering on the link data in order to dynamically "discover" and show only the interesting links. In this way, the graph is kept readable and the visualization can be also enriched with the names of the discovered genes that make up the starting

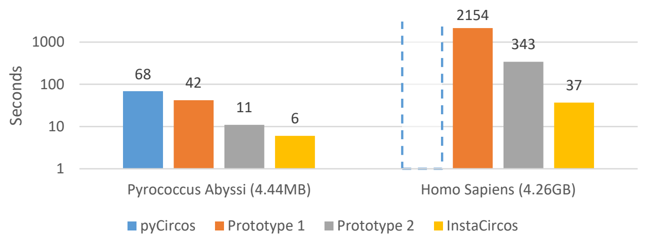


Fig. 6. Performance comparison: execution time for small genome (left) and large genome (right) visualization

and ending places, as obtained from the selected links (please note that Figure 4 shows only a very limited number of sample links in order to give an idea of their representation).

In the final version of the tool, we will provide additional tabs containing all the selectable options and parameters to dynamically create new plots, update the current one by removing or adding new tracks, loading custom data, etc.

VI. Development Stages and Performance Evaluation

In this section we will briefly analyze the performances of InstaCircos in terms of execution time required to generate and display a graph, both in the case of a small (Pyrococcus Abyssi, GBFF file size 4.44MB, excluding links) and a very large (human genome, GBFF file size 4.26GB, excluding links). In particular, we will compare the performances with one of the most well known and used Python packages for static circular plot generation, pyCircos. In order to understand the impact of specific implementation choices, we will consider the different evolution of InstaCircos by discussing its key implementation stages and performances. Tests were executed on a standard notebook with a x64-based CPU @2.70GHz, 12GB RAM and a 1TB hard drive.

The first stage (that we will call "Prototype 1") has been developed as a static (non-interactive) application similar to pyCircos, using Pandas and Matplotlib, the comprehensive library for creating static visualizations in Python. The first aim of the application was, in fact, to speed up the plot creation of large genome file, such us the human genome. In Prototype 1, and similarly to what is done in pyCircos and in most state of the art, data was managed in main memory. The plot was generated with Matplotlib functions, similarly

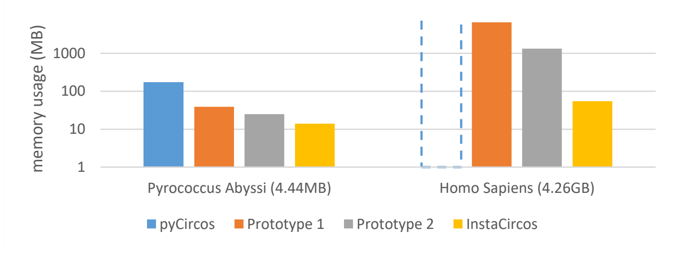


Fig. 7. Server-side memory usage comparison: memory usage for small genome (left) and large genome (right) visualization

to how the plot creation on pyCircos is handled, but, unlike the latter, only the strictly necessary data was parsed and stored in Pandas DataFrame, making it possible to visualize big genomes. As shown in Figure 6, the performances of pyCircos and Prototype 1 in creating the same kind of plot, are the following: 68 seconds and 42 seconds, respectively. Generating the same plot for the human genome is impossible with pyCircos, due to a memory error. With Prototype 1 the plot was successfully generated in 2154 seconds.

Since execution time for big genomes was still a down point, we worked on Prototype 2 specifically on the data management part. All the information was stored and retrieved in a database structured as described in the paper, significantly reducing the memory usage, as well as the performance time which dropped down to 11 seconds for the Pyrococcus Abissy and to 343 seconds for the human genome.

Finally, Prototype 2 evolved in the InstaCircos version we present in this paper. After a detailed profiling of the code, it was clear most of the execution time was due to the plot saving and the feature visualization. Further improvements were made by optimizing the visualization functions, which in state of the art tools such as pyCircos are typically called for each and every feature selected. Once the execution time became acceptable for an interactive interface, the tool was given an interactive web interface, leading to InstaCircos current implementation, As shown in Figure 6, these further changes dropped the execution time of both the feature plotting and the plot saving, for a total execution time of 6 seconds for the small and 37 seconds for the human genome.

Speaking about server-side memory usage, the pyCircos library on the Pyrococcus Abissy genome takes up to 172 MB. Our Prototype 1 reduces it to 39MB, and the memory usage drops down to 25MB and 14MB for Prototype 2 and current InstaCircos implementation, respectively, thanks to the adoption of the database. For generating the human genome plot the differences are even more noticeable: while pyCircos goes out of memory, InstaCircos allows its management in as little as 55 MB (compared with 6.43GB and 1.3GB of Prototype 1 and 2).

VII. Conclusions and Future Work

Genomic and, more generally, scientific research is in constant need for tools allowing researchers to view and explore large quantities of data with effective visual forms, so to foster new discoveries. The InstaCircos project is born from the scientific collaboration between Big Data Analytics and Bioinformatics researchers and enables the generation of circular layout graphs (particularly advantageous for the immediate feedback and for the visual impact of the result) that are: (a) efficient in their generation and interactive, allowing users to customize and explore the relationships between the elements in different positions of the graph; (b) easy to use requiring no installation or coding competencies; (c) applicable to very large genomes without any memory requirement. No current state of the art tool we are aware of satisfies all the above requirements.

The project is still in development and will be made publicly available as a web application once finished, as we did in the past for other genomic exploration tools [15]. The current prototype version already shows promising performances both in terms of execution time and memory requirements. In the future, we will: (i) select and add to the database a representative number of additional notable genomes to be explored; (ii) consider additional features, including richer UI with additional options and parameters to dynamically manage the plots and select user-defined portions of the data, and support for custom user-provided genome storing and visualization.

References

[1] M. Krzywinski, J. Schein, I. Birol, J. Connors, R. Gascoyne, D. Horsman, S. Jones, and M. Marra, "Circos: an information aesthetic for comparative genomics," *Genome research*, vol. 19, pp. 1639–45, 07 2009.

[2] "Circos," http://circos.ca/.

[3] H. Zhang, P. Meltzer, and S. Davis, "Rcircos: An r package for circos 2d track plots," *BMC bioinformatics*, vol. 14, p. 244, 08 2013.

[4] "RCircos: Circos 2D Track Plot," http://cran.r-project.org/web/packages/RCircos.

[5] Y. Hu, C. Yan, C.-H. Hsu, Q.-R. Chen, K. Niu, G. A. Komatsoulis, and D. Meerzaman, "Omiccircos: A simple-to-use r package for the circular visualization of multidimensional omics data," *Cancer Inform*, vol. 13, pp. 13–20, 01 2016.

[6] "OmicCircos: High-quality circular visualization of omics data," http://bioconductor.org/packages/release/bioc/html/OmicCircos.html/.

[7] N. Darzentas, "Circoletto: visualizing sequence similarity with Circos," *Bioinformatics*, vol. 26, no. 20, pp. 2620–2621, 08 2010.

[8] "Circoletto," http://omictools.com/circoletto-tool/.

[9] "Python Modules for Circos Plot," http://github.com/KimBioInfoStudio/PyCircos/.

[10] D. Otasek, C. Pastrello, A. Holzinger, and I. Jurisica, *Visual Data Mining: Effective Exploration of the Biological Universe*. Springer Berlin Heidelberg, 2014, pp. 19–33.

[11] I. Kuznetsova, A. Filipovska, O. Rackham, A. Lugmayr, and A. Holzinger, "Circularized visualisation of genetic interactions," in *Proc. of WWW Companion*, 2017, p. 225–226.

[12] "NCBI GenBank," http://www.ncbi.nlm.nih.gov/genbank/.

[13] W.-H. Cheong, Y.-C. Tan, S.-J. Yap, and K. P. Ng, "Clico fs: An interactive web-based service of circos," *Bioinformatics (Oxford, England)*, vol. 31, 07 2015.

[14] "NCBI GenBank," http://clicofs.codoncloud.com/.

[15] V. Lomonaco, R. Martoglia, F. Mandreoli, L. Anderlucci, W. Emmett, S. Bicciato, and C. Taccioli, "Ucbase 2.0: Ultraconserved sequences database (2014 update)," *Database: the journal of biological databases and curation*, vol. 2014, 2014.