Connecting Integrated Genome Browser to a huge genome database using its open API solves one problem and creates another

Ann Loraine[1,2], Nowlan Freese[2], Jaya Sravani Sirigineedi[2], Sameer Shanbhag[2]

[1] Contact Email: aloraine@charlotte.edu

[2] University of North Carolina at Charlotte, 9201 University City Blvd, Charlotte, NC 28223

Project Website: https://bioviz.org/

Source Code: https://bitbucket.org/lorainelab/integrated-genome-browser, https://bitbucket.org/lorainelab/genome-dashboard, https://bitbucket.org/lorainelab/bioviz, https://bitbucket.org/lorainelab/bioviz-playbooks

Integrated Genome Browser (IGB, pronounced "ig-bee") is a fast, feature-rich, open-source desktop genome browser used by thousands of researchers in many countries over many years to explore and analyze genomic data. To support our user audience, we maintain data delivery Web sites called "IGB Quickloads" that supply IGB with around 60 different reference genome assemblies, including many not yet available in other browsers, such as tardigrade and many plant species. IGB can open and display genome assembly files provide by users, but users can avoid this inconvenient work. To show which assemblies are already available, we created a searchable genome dashboard Web application that displays the available genome assemblies. Selecting an assembly in the dashboard prompts IGB to show that assembly. However, we are finding it increasingly difficult to update these Quickload sites with new assemblies. Fortunately, many large genome informatics groups now offer robust computational access to their data. Using these computational access points, we should be able attempting to import assemblies into IGB. To test this idea, we developed a new IGB version that consumes and displays data from one such resource, a JSON-emitting REST API (application programming interface) from the University of California at Santa Cruz Genome Browser informatics group. Now available as an "early access" version at the BioViz.org Web site, this new IGB version can display 220 assemblies visible in the Santa Cruz Browser, along with dozens of annotation and data tracks. This wealth of data has given us a new problem to solve. The API provides information like track names and data file formats (e.g., "psl" and "wig"), but almost nothing about the computational or laboratory experiments that produced the data. Thus, our group faces a new form of a long-standing question in bioinformatics: how do we categorize and label information so that computer programs and people can understand and use it?