MoMI-G Documentation

Release 1.0.0

Toshiyuki Yokoyama

Jul 15, 2023

Getting Started:

CHAPTER 1

MoMI-G Quick Start

MoMI-G consists of two parts, frontend and backend (MoMI-G_backend). MoMI-G is available from source code. MoMI-G backend can be built from source courde, but we recommend to use docker image for reducing time for building backend server and setting up test environment.

1.1 Frontend

First, you need to install git, node, and npm or yarn. See here for installation instructions.

Then, you can build MoMI-G with

```
$ git clone https://github.com/MoMI-G/MoMI-G
$ cd MoMI-G
$ yarn
```

And access to http://localhost:3000/. The demo shows CHM1, a human hydatidiform mole cell line dataset from backend server of MoMI-G that MoMI-G developer serves.

1.2 Backend

If you want to run the backend server on your laptop or workstation, we recommend using docker image of MoMI-G backend server. As an example, we provide CHM1 chr21 and simulated reads dataset. For running a demo, modify package.json and run docker container which includes backend server.

First, you have to install docker. See here for installation instructions.

Then, you have to change the backend URL from the MoMI-G public server to localhost, and run the backend server.

```
$ sed -e "s/\"target/\"target_/g"  -e "s/\_target/target/g" -i.bak package.json
$ docker run --init -p 8081:8081 momigteam/momig-backend # Run it on another shell.␣
↪It takes a little long time -- please wait.
$ yarn start
```

You can start docker container by docker-compose up instead of docker run.

1.3 Custom Backend For Latest VG

If you want to use the latest version of vg, you have to change the version of vg used on backend server due to the compatibility of xg binary format. Therefore, we recommend using a custom build of docker binary.

First, you modify the first line of `Dockerfile.backend` to set vg's version you used.

```
FROM quay.io/vgteam/vg:v1.14.0 as build

# frontend container
FROM momigteam/momig-backend

COPY --from=build /vg/bin/vg /vg/bin/

EXPOSE 8081

CMD ["./graph-genome-browser-backend", "--config=static/config.yaml", "--
↪interval=1500000", "--http=0.0.0.0:8081", "--api=/api/v2/"]
```

If you use `vg:v1.6.0-213-gc0c19fe5-t126-run`, then the first line should be modified as `FROM quay.io/vgteam/vg:v1.6.0-213-gc0c19fe5-t126-run as build`.

After that, you build the custom backend and run the built backend server instead of official docker image.

```
$ docker build -t momig-custom-backend -f Dockerfile.backend .
$ docker run --init -p 8081:8081 momig-custom-backend
```

1.4 How to use your own custom data

1. Git clone from https://github.com/MoMI-G/MoMI-G.
2. `mkdir static` on the directory MoMI-G clones.
3. Put your XG data in the `static` folder like `static/data.xg`.
4. Add configure file on `static/config.yaml`.
5. Modify `Docker.backend` file as follows.

```
FROM quay.io/vgteam/vg:v1.17.0 as build

# backend container
FROM momigteam/momig-backend

COPY --from=build /vg/bin/vg /vg/bin/
COPY static/data.xg /vg/static/
COPY static/config.yaml /vg/static/

EXPOSE 8081

CMD ["./graph-genome-browser-backend", "--config=static/config.yaml", "--
↪interval=1500000", "--http=0.0.0.0:8081", "--api=/api/v2/"]
```

6. Build backend server and run.

```
$ docker build -t momig-custom-backend -f Dockerfile.backend .
$ docker run --init -p 8081:8081 momig-custom-backend
```

CHAPTER 2

Why MoMI-G?

There are many genome browsers or SV visualization tools; why does MoMI-G exist?

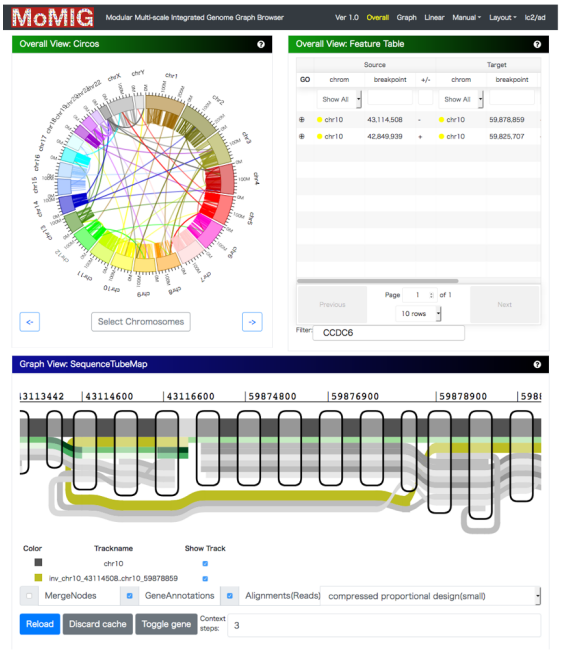
To our knowledge,MoMI-G is the only SV visualization tool that satisfies the following conditions:

- (1) allows visualization of (possibly distant) multiple intervals;
- (2) displays SVs that span multiple intervals;
- (3) displays SVs at varying scales, i.e., chromosome, gene, and nucleotide scales;
  - 4a. the chromosome scale view can show the distribution of SVs on one or more chromosomes;
  - 4b. the gene scale view can show annotations such as exon/intron structures and repeats;
  - 4c. the nucleotide scale view can show nucleotide-level alignments, in particular, read alignments that correspond to both alleles of heterozygous SVs are shown simultaneously;
- (5) allows users to manually inspect hundreds of SVs.

MoMI-G provides a graph-based view that displays a genome with branches and alignments on them. Users can filter, visualize with genomic annotations, and inspect SVs with read alignments.

https://github.com/MoMI-G/MoMI-G

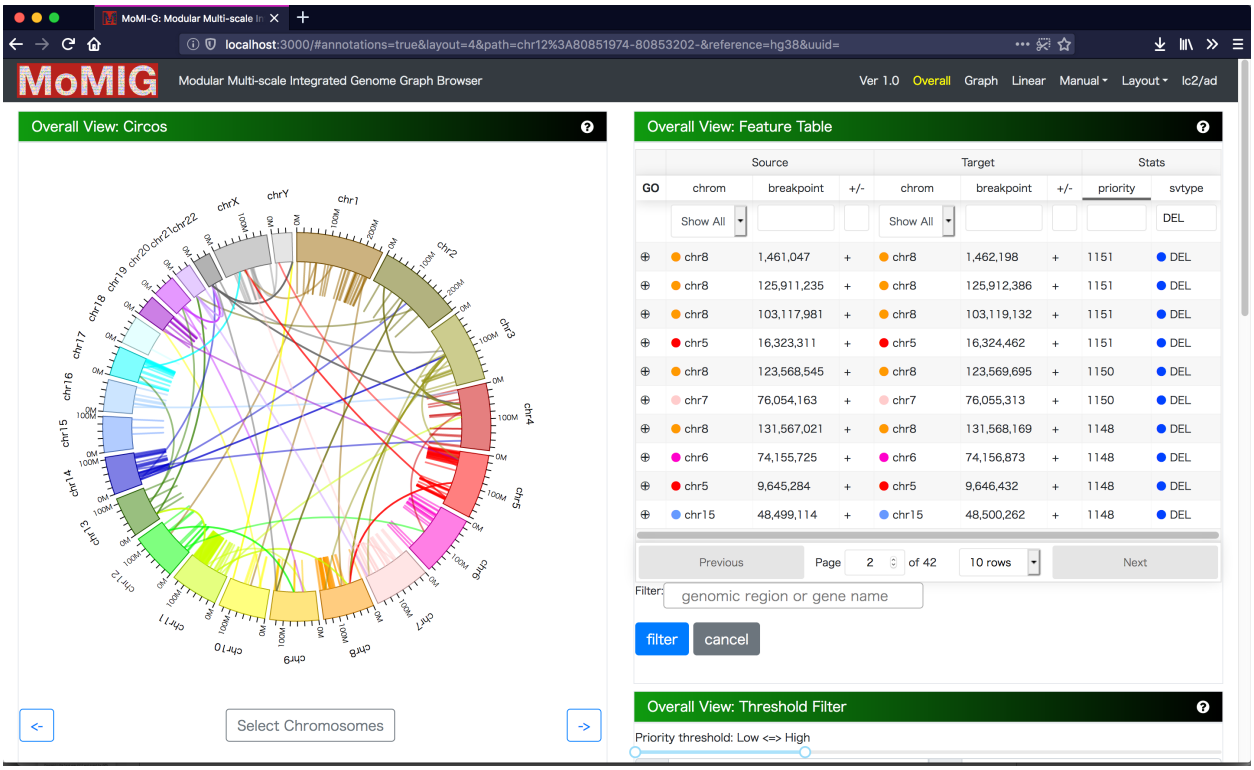
- ☑ Compare between genomes
- ☑ Visualize nested SVs
- ☑ Navigate for every SV
- ☑ Inspect the authority of SV



CHAPTER 3

View Modules

3.1 Circos Plot / Feature Table



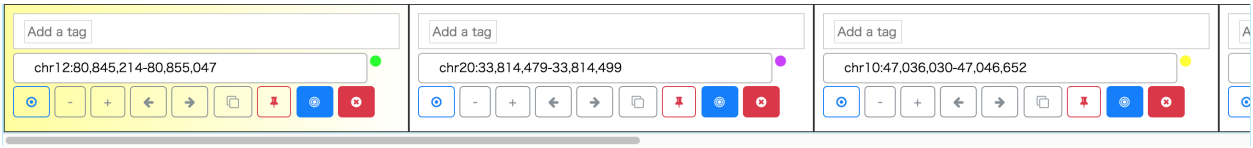
Manual screening for SV candidates is still important because of high false positive rate of SV calling, but it is time-consuming so that SVs should be filtered by certain criteria for manual screening. There are two view components to select SVs from candidates. The first view components is Circos, which aligns chromosomes as a circular layout. Thick arcs are chromosomes, and a line connecting to thick arcs indicates a SV. Because Circos enables us to choose

and rotate chromosomes, we can focus on the subset of chromosomes. Circos is a suitable for visualizing the distribution of SVs and inter-chromosomal variations as lines. Second, Feature Table shows chromosome names, coordinates, and strand, enabling us to select SVs. We can sort and filter Feature Table by SV type, coordinates, or gene name. We can select a SV from both components.

3.2 Threshold Filter

Threshold Filter has two use cases. First, one can toggle checkboxes to select whether to show inter-chromosomal SVs and/or intra-chromosomal SVs. Second, one can filter SVs by using a slider based on the custom priority (possibly given by SV callers) of each SV.

3.3 Interval Card Deck

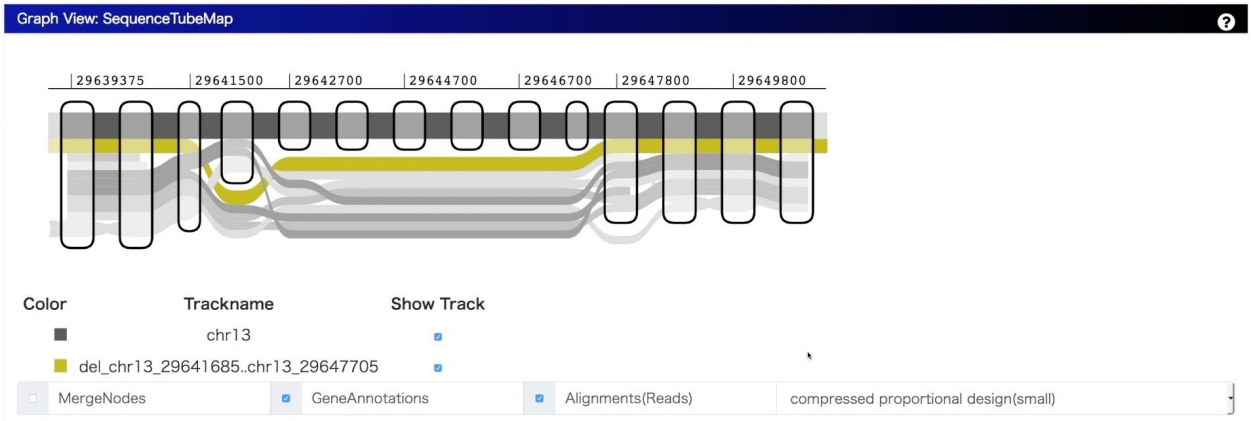


After you select SVs using Feature Table or Circos Plot, the listed variants are stacked on Interval Card Deck at the bottom of the window. In Interval Card Deck, intervals are displayed as cards, and the interval of the top (leftmost) card of the deck is shown on SequenceTubeMap. Each card can be dragged, and the order of cards can be changed. If one double-clicks on a card, the card moves to the top of the deck. Also, a card can be locked to avoid unintended modification or disposal, and the gene name can be input with autocompletion for specifying the interval of a card.

Shortcut keys

- Option/Alt + d -> Delete the head of the deck
- Option/Alt + s -> Keep the card, and move to the bottom of the deck

3.4 SequenceTubeMap

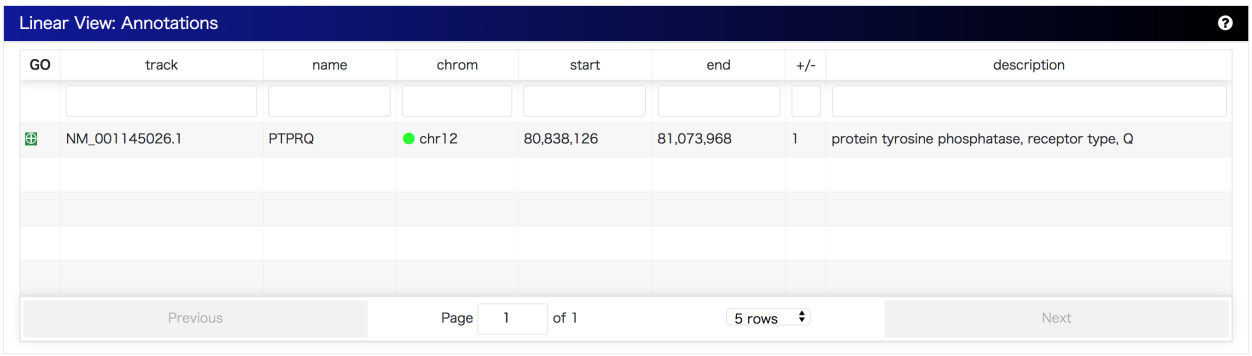


We integrate SequenceTubeMap into MoMI-G with modifying the original implementation so that it can visualize a variation graph converted from SVs for showing the difference between a reference genome and a personal genome. Because there are many types of paths in variation graph, we categorized them for assigning different design as follows:

- Chromosome Path (thick, grayscale): A chromosome path is a chromosome in a reference genome; walking on the path provides us a full nucleotide sequence.
- Variant Path (colored): A variant path with a variation name represents a personal genome.
- Gene Path (thin, colored): A gene path is for gene annotation. Exons are shown as a path with a darker color, and introns are shown as that with a lighter color.
- Annotation Path (thin, grayscale): An annotation path is for user-definable bigBed annotations such as repeats. If you have a GFF3 or BED file, you can easily convert into bigBed file.
- Read Alignment (thin, grayscale): Read alignments aligned on the graph or lifted over from the original alignments to a reference genome.

A sequence graph as a bi-directed graph composed of a set of multiple DNA sequences as nodes and the corresponding end-to-end connections as edges has the ability to describe duplications, indels and inversions as loops or branching and merging against a reference genome.

3.5 Annotations



Gene annotations retrieved from Ensembl or names of the region described in bigbed format are listed on Annotation Table. Annotation table shows all annotations that are displayed on the SequenceTubeMap. Moreover, annotations can be downloaded as a BED file.

Ideally, MoMI-G provides annotations on variation graphs. However, annotations available in public databases are for the linear reference genome. MoMI-G can display annotations in bigWig/bigBed formats. In particular, for human reference genomes, GRCh19/hg37 and GRCh38/hg38, MoMI-G provides an interface for retrieving Ensembl gene annotations from the Ensembl SPARQL endpoint (Jupp et al. 2014) via SPARQList REST API (https://github.com/dbcls/sparqlist). The orientation of genes is shown in the legend of SequenceTubeMap. Further, if one clicks on a gene name, the website of the gene information in TogoGenome (Katayama et al. 2019) opens.

3.6 Linear Genome Browser

To provide a compatible view of a selected genomic region, we integrated Pileup.js (Vanderkam et al. 2016) into MoMI-G.

CHAPTER 4

Custom Input

This is an instruction for using custom input on MoMI-G. When you want to employ custom dataset, you have to do following procedure; see references.

1. Convert SV files into a vg format; use MoMI-G tools or generate variation graphs on their own.
2. Put annotation files you want to visualize on the `static/` directory.
3. Modify configuration file written in YAML.

4.1 MoMI-G data files

The list of formats that are accepted by MoMI-G is following. You should put them on the `static/` directory and specify the file name on the YAML configuration file.

File type	Extension	Description	Softwares
A succinct index of variation graphs	.xg	Variation graphs displayed in MoMI-G. (required)	vg
Graphical alignment/map	.gam	Read alignment.	vg
Comma-separated values	.csv(.pcv)	SV list for chromosome-scale view.	MoMI-G tools
Browser extensible data	.bed	Used for converting gene names to genomic intervals.	
Compressed binary indexed BED	.bb	Annotation tracks.	bedToBigBed
Compressed binary indexed wiggle	.bw	Annotation tracks.	bedGraphToBigWig

If you want to use bed/gff/gtf files for custom tracks on SequenceTubeMap, you have to manually convert them into bigbed format.

CHAPTER 5

Parameter

To get options, run

```
$ docker run momigteam/momig-backend ./graph-genome-browser-backend --help
```

You can specify parameters like this.

```
$ docker run momigteam/momig-backend ./graph-genome-browser-backend --intervals=100000
```

5.1 Options

config <string="config.yaml"> A path to a configuration file.

http <string="127.0.0.1:8081"> A host and port.

threads <int=1> (UNUSED) Threads per process.

tmp <string="./tmp"> Cache folder for reducing computational time for subgrph retrieving by storing json generated from vg. Caches can be permanent because subgraphs are stable unless you update a variation graph or read alignments. Remove all caches when you update the data.

static <string="./static"> Static files for providing csv file or bigWig/bigBed files. All files that serve as static files should be located in the static directory.

build <string="./build"> A directory that includes MoMI-G frontend website. MoMI-G backend serves MoMI-G frontend; otherwise, you can directly serve MoMI-G frontend HTML/CSS/JS files via a reverse proxy such as nginx.

rocksdb <string="./rocksdb"> (OBSOLETE) This option remains for backward compatibility.

api <string="/api/v1/"> A prefix of API endpoint. It is used for redirect to the static file.

interval <int=50000> A maximum threshold of an interval against the reference genome. When a request from client exceeeds the threshold, MoMI-G server returns an error.

CHAPTER 6

Configuration File

MoMI-G backend requires a configuration file written in YAML which describes paths for datasets and the configuration about a reference genome.

6.1 Overview

```yaml
bin:
  vg: "vg"
  vg_tmp: "vg"
  graphviz: "dot"
  fa22bit: "faToTwobit"
  bigbed: "bedToBigBed"
reference:
  chroms: "static/GRCh.json"
  data:
    - name: "hg19"
      features:
        - name: ""
          url: ""
          chr_prefix: ""
    - name: "hg38"
      features:
        - name: ""
          url: ""
          chr_prefix: ""
data:
  - name: ""
    desc: ""
    chr_prefix: ""
    ref_id: ""
    source:
      xg: ""
      twobit: ""
```

(continues on next page)

(continued from previous page)

```
      gamindex: ""
    features:
      - name: ''
        url: ''
        chr_prefix: ''
    static_files:
      - name: ''
        url: ''
        viz: ''
```

bin <string> A relative/absolute path for the binary. At least, the path for vg is required. Values for fa22bit, bigbed and graphviz are optionally and currently unused. vg_tmp is the same as the vg, but it is used for processing temporary uploaded files. Sometimes temporary uploaded files are stored in the separated directory, we can specify

reference <section> This section is used for describing a reference genome such as hg19 or hg38. You can write multiple reference genomes for sharing the same section between samples.

reference :> chroms <string="*.json"> A relative/absolute path for chromosomal information. The json file includes the color and length for every chromosome. You can use the preset chromosome color scheme attatched the MoMI-G backend for human dataset; otherwise you can manually write json file.

reference :> data :> features :> url <string="*.[bed|gff|gtf]"> A relative/absolute path for genomic features which are indexed with the name of the feature. All intervals of genes/features written in this section are indexed, so you can query the gene/feature name for specifying the interval of it.

reference :> data :> features :> chr_prefix, data :> chr_prefix <string="" || string="chr"> There are a few ways for representing chromosome 1. One might write "chr1"; the other might write "1". Specify prefix; then, the differences in prefix are implicitly handled.

data <section> This section is used for representing samples. You can write multiple data sources; but currently the first item of the data section is visualized.

data :> chr_prefix <string> There are a few ways for representing chromosome 1. One might write "chr1"; the other might write "1". Specify prefix; then, the differences in prefix are implicitly handled.

data :> ref_id <string="hg18"> Ref_id is corresponding to the `reference :> data :> name`. You can specify the reference genome of the given data for fetching genomic annotation from SPARQL endpoint.

data :> source :> twobit <string="*.2bit"> (OBSOLETE) `.2bit` files are used in pileup.js for rendering reference nucleotides, but you should list twobit file on `data :> static_files`. This option remains for backward compatibility.

data :> source :> gamindex <string="*.gam.index"> Read alignments against genome graphs. Note that you should specify a **.gam.index* file instead of **.gam* file.

data :> features <section> All features written in this section are displayed on the custom track of SequenceTubeMap. The detail is the same as *reference :> data :> features*.

data :> features :> url <string="*.[bb|bw]"> MoMI-G supports bigWig and bigBed format as data sources.

data :> static_files <section> All files written in this section are displayed on `pileup.js` used in the MoMI-G.

data :> static_files :> url <string="."> All files in this section should be a relative path from `static` folder specified in MoMI-G backend parameter or absolute URL started from `http://` or *https://*.

data :> static_files :> viz <string="[twobit|bigbed|variants|bam]"> Specifying the type of static file is required to be displayed on `pileup.js`. The following table describes the corresnponding viz keywords for supporting extensions.

extension	vis	pileup.js
.2bit	twobit	pileup.viz.genome()
.bb	bigbed	pileup.viz.genes()
.vcf	variants	pileup.viz.variants()
.bam	bam	pileup.formats.bam()

6.2 Example

```
reference:
  chroms: "static/GRCh.json"
  data:
    - name: "hg19"
      features:
        - name: 'gene_annotation'
          url: "test/gencode.v25.basic.annotation.Y.bed"
          chr_prefix: "chr"
    - name: "hg38"
      features:
        - name: 'gene_annotation'
          url: "test/gencode.v25.basic.annotation.Y.bed"
          chr_prefix: "chr"
data:
  - name: "na12878"
    chr_prefix: "chr"
    ref_id: "hg38"
    source:
      xg: "x5.xg"
    features:
      - name: 'ensgene'
        url: "test/ensgene.bb"
        chr_prefix: "chr"
      - name: 'repeat_annotation'
        url: "test/repeats.bb"
        chr_prefix: "chr"
    static_files:
      - name: 'Reference'
        url: 'https://www.biodalliance.org/datasets/hg38.2bit'
        viz: 'twobit'
      - name: 'Genes'
        url: 'http://www.biodalliance.org/datasets/ensGene.bb'
        viz: 'bigbed'
      - name: 'Variants'
        url: './samples/remapped_NA12878.sorted.vcf'
        viz: 'variants'
```

6.3 Using vg docker image

You can use docker image of vg instead of binary version of vg. If you want to use docker, you should specify the docker image on the `bin` section like following.

```
bin:
  vg: "/usr/local/bin/docker run --rm --memory-reservation 2G -i -v tmp:/tmp quay.io/
↪vgteam/vg:v1.5.0-581-gc51e1acb-t67-run vg"
  vg_tmp: "/usr/local/bin/docker run --rm --memory-reservation 2G -i -v /tmp:/tmp␣
↪quay.io/vgteam/v1.6.0-60-g515433ff-t120-run vg"
```

We recommend to limit the memory the container can use, but it does not guarantee that the container doesn't exceed the limit because it is a soft limit.

CHAPTER 7

Custom Layout

MoMI-G supports the custom layout of view modules; users can customize the view if they are not satisfied with the preset views.

You can modify the layout of view modules on GUI; You can modify the layout of view modules on the web browser, but you cannot save the custom layout because it is volatile. Alternatively, we recommend configuring the layout file manually.

See `src/dashboard/preset.json`.

```
{
  "Default": {
    "rows": [
      {
        "columns": [
          {
            "className": "col-md-6 col-sm-6 col-xs-6",
            "widgets": [{ "key": "CircosWidget" }]
          },
          {
            "className": "col-md-6 col-sm-6 col-xs-6",
            "widgets": [
              { "key": "ThresholdWidget" },
              { "key": "FeatureTableWidget" }
            ]
          }
        ]
      },
      {
        "columns": [
          {
            "className": "col-md-12 col-sm-12 col-xs-12",
            "widgets": [
              { "key": "TubeMapWidget" },
              { "key": "AnnotationWidget" }
            ]
```

(continues on next page)

(continued from previous page)

```
            }
          ]
        }
      ]
    }
}
```

The layout is on the dazzle. You can select view modules from the following list.

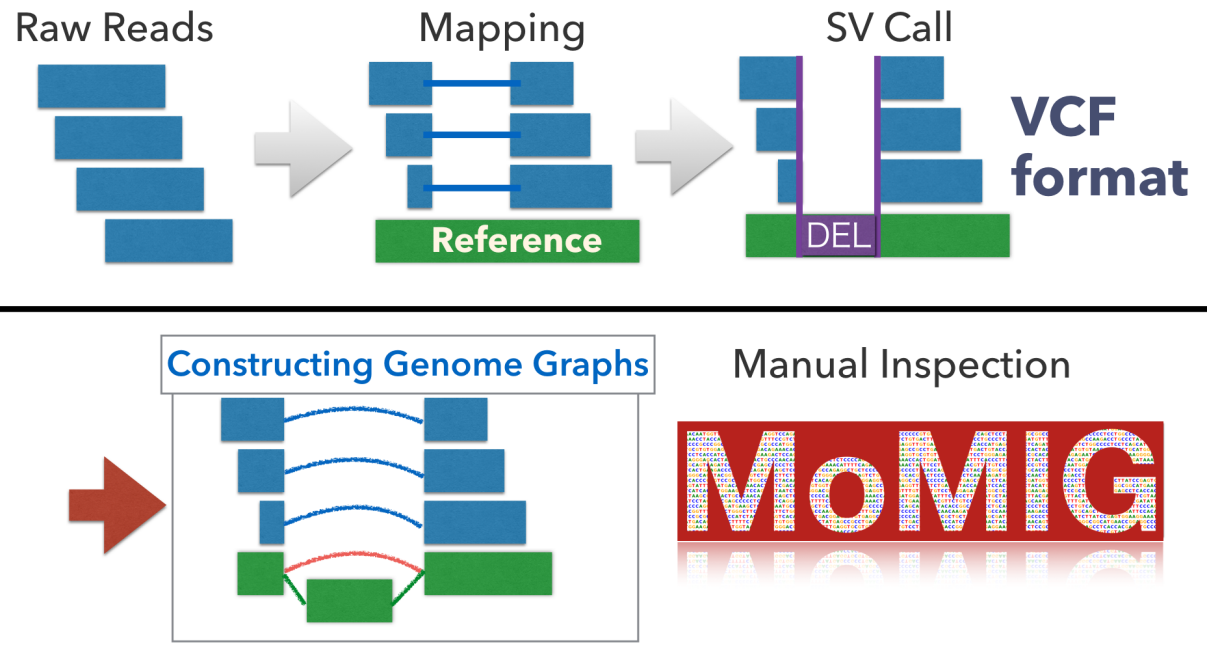
Module name	View module	Description
CircosWidget	Circos Plot	Observe the distribution of SVs
FeatureTableWidget	Feature Table	Select and filter SVs
ThresholdWidget	Threshold Filter	Filter SVs
TubeMapWidget	SequenceTubeMap	Visualize a subgraph of a variation graph
AnnotationWidget	Annotation Table	Show annotations
PileupWidget	Linear Browser	Visualize a linear genome with annotations
PathRegionWidget	Interval Card Deck	Switch between the intervals

After modifying `src/dashboard/preset.json`, you should rebuild MoMI-G frontend.

CHAPTER 8

MoMI-G tools

The MoMI-G package includes a set of scripts (MoMI-G tools) that converts a VCF file into the variation graph format.



If you want to run MoMI-G with your own dataset, use our custom scripts `scripts/vcf2xg.sh` to generate pcf and xg dataset. It requires VG, ruby, bash, and samtools.

Software	Dataset	Supported SV type
Sniffles	for PacBio/Nanopore	INV/DEL/DUP/TRA/(INS)
10X LongRanger	for 10X	INV/DEL/DUP/TRA
SURVIVOR	for merging SV calls	INV/DEL/DUP/TRA

```
$ bash vcf2xg.sh test.vcf test_output /bin/vg hg[19|38]
```

Otherwise, you can specify your own reference file as follows.

```
$ bash vcf2xg.sh test.vcf test_output /bin/vg /data/hs37d5.fa
```

You can use singularity instead.

```
$ singularity -s run docker://momigteam/momig-tools:master test.vcf test_output vg␣
↪hg38
```

After that, these files are required to be mounted on `static/` folder. Also, you should modify `config.yaml` and *Dockerfile.backend* in MoMI-G directory.

- **static/**
  - config.yaml: a configuration file
  - *.xg: an index of a variation graph, generated by [vg](https://github.com/vgteam/vg)
  - *.pcf: a pair of coordinate format file: required to display variants on Feature Table or Circos Plot.
  - *.gam(optional): read alignments on the graph
  - *.gam.index(optional): index of gam

This is an example of config.yaml and Dockerfile. `\*` is just an example and you need to replace it to the actual file name.

```
bin:
  vg: "vg"
  vg_tmp: "vg"
  vg_volume_prefix: ""
  graphviz: "dot"
  fa22bit: "faToTwoBit"
  bigbed: "bedToBigBed"
reference:
  chroms: "static/GRCh.json"
  data:
    - name: "hg19"
      features:
        - name: 'gene_annotation'
          url: "static/gencode.v27lift37.basic.annotation.gff3"
          chr_prefix: "chr"
    - name: "hg38"
      features:
        - name: 'gene_annotation'
          url: "static/gencode.v27.basic.annotation.gff3"
          chr_prefix: "chr"
data:
  - name: "data"
    desc: "2019/12/19"
    chr_prefix: "chr"
    ref_id: "hg38"
    source:
      xg: "static/\*.xg" # To be rewritten
      csv: "static/\*.pcf" # To be rewritten
#      gam: "static/b.gam"
#      gamindex: "static/b.gam.index"
```

(continues on next page)

(continued from previous page)

```
    features: []
    static_files: []
```

```
# Specify the version you used to build xg index.
FROM quay.io/vgteam/vg:v1.14.0 as build

# frontend container
FROM momigteam/momig-backend

COPY --from=build /vg/bin/vg /vg/bin/
# Move these files into /vg/static/ folder.
COPY static/\*.xg /vg/static/
COPY static/\*.pcf /vg/static/
COPY static/config.yaml /vg/static/
EXPOSE 8081

CMD ["./graph-genome-browser-backend", "--config=static/config.yaml", "--
↪interval=1500000", "--http=0.0.0.0:8081", "--api=/api/v2/"]
```

If you use the later VG (version >= 10), `gam.index` file is no longer used. Please use `sorted.gam.gai` instead (bam2gam.sh generates `gam.gai` file).

- **static/**
  - config.yaml: a configuration file
  - *.xg: an index of a variation graph, generated by [vg](https://github.com/vgteam/vg)
  - *.pcf: a pair of coordinate format file: required to display variants on Feature Table or Circos Plot.
  - *.sorted.gam(optional): read alignments on the graph
  - *.sorted.gam.gai(optional): index of gam

Then, run the MoMI-G backend.

```
$ docker build -t momig-custom-backend -f Dockerfile.backend .
$ docker run --init -p 8081:8081 -v `pwd`/static:/vg/static momig-custom-backend
```

At last, run the MoMI-G frontend.

```
$ sed -e "s/\"target/\"target_/g"  -e "s/\_target/target/g" -i.bak package.json
$ yarn
$ yarn start
```

CHAPTER 9

Project Information

9.1 Citation

Yokoyama, T.T., Sakamoto, Y., Seki, M., Suzuki, Y., Kasahara, M. MoMI-G: modular multi-scale integrated genome graph browser. BMC Bioinformatics 20, 548 (2019) doi:10.1186/s12859-019-3145-2.

9.2 Funding

This work was supported in part by Information-technology Promotion Agency, Japan (IPA), Exploratory IT Human Resources Project (The MITOU Program) in the fiscal year 2017 and in part by JSPS KAKENHI (Grant Number, 16H06279).

9.3 Changelog

- 2017.08.09 Ver 0.1 (alpha)
- 2017.09.26 Ver 0.2 (deploy CHM1 graph)
- 2017.12.12 Ver 0.3 (beta)
- 2017.12.19 Ver 0.4 (add gene annotations)
- 2018.01.02 Ver 0.5 (add English descriptions)
- 2018.01.11 Ver 0.6 (fix design)
- 2018.01.28 Ver 1.0RC
- 2018.02.10 Ver 1.0
- 2019.01.06 Ver 1.0 Rev.1

9.4 License

MoMI-G is licensed under MIT License.

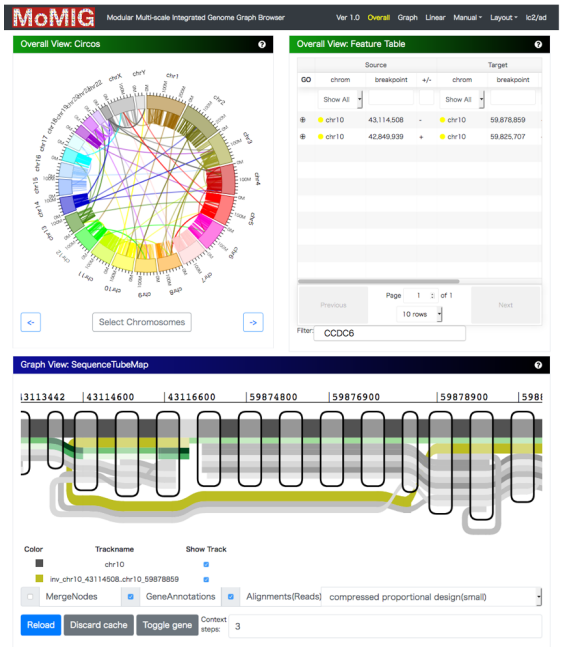
CHAPTER 10

Modular Multi-scale Integrated Genome Graph Browser

MoMI-G is a genome graph browser to visualize SVs on the variation graph, that provides a graph-based view that displays a genome with branches and alignments on them. Users can filter, visualize with genomic annotations, and inspect SVs with read alignments.

https://github.com/MoMI-G/MoMI-G

- ☑ Compare between genomes
- ☑ Visualize nested SVs
- ☑ Navigate for every SV
- ☑ Inspect the authority of SV



10.1 Demo

- **Youtube**

  - Short Demo
  - Short Demo2
  - Long Demo
- Demo Page

10.2 Install

We recommend to build the latest commit from github. If you encounter any issues, please report them using the github issues page.

```
$ git clone https://github.com/MoMI-G/MoMI-G
$ cd MoMI-G
$ yarn
```

For using your custom data.

10.3 Support

- For releases, see Changelog.
- To discuss with other users or post questions, you can use gitter.
- For bugs and feature requests, please use the issue tracker.
- For contributions, visit MoMI-G on github.

10.4 Citation

Yokoyama, T.T., Sakamoto, Y., Seki, M., Suzuki, Y., Kasahara, M. MoMI-G: modular multi-scale integrated genome graph browser. BMC Bioinformatics 20, 548 (2019) doi:10.1186/s12859-019-3145-2.

10.5 License

MIT