Benchmarking Di4

vs. Di3, BEDTools, and BEDOPS

Vahid Jalili

4-12-2018

Contents

1. Preamble

In the following, we explain the exact steps to run Di4, Di3, BEDTools, and BEDOPS.

Note

Di4 is under active development, and we strive to deliver a highly portable, holistic, generic, extendable, and performant framework to support interval-based genomic data indexing and querying. Di4B (a layer on top of Di4 that extends Di4 for genomic domain applications) and Di4BCLI (a layer on top of Di4B that provides command-line user interactions) are also under active development. Accordingly, there could be bugs in portability and functionality of Di4, Di4B, and/or Di4BCLI. Once we spot or be notified of any possible bugs, we will do our best to resolve them. You may report bugs on Di4's GitHub page at the following link:

https://github.com/Genometric/Di4/issues

Latest Di4 documentation is available from the following page:

https://genometric.github.io/Di4/docs

Di3 is the predecessor of Di4; hence, we fix in Di4 any possible bugs in Di3's functionality and/or portability.

2. Datasets

The data used for benchmarking are public data and are real genomic datasets (no simulation or artificial manipulation) which are downloaded from the ENCODE repository (https://www.encodeproject.org). The filenames are not changed so that each file can be traced back to its source. Some of the source files are available at the following links:

- http://hgdownload.cse.ucsc.edu/goldenpath/hg19/encodeDCC/wgEncodeUwDnase/
- http://genome.ucsc.edu/cgi-bin/hgFileUi?db=hg19&g=wgEncodeUwDnase

For benchmarking purpose, we organize these data in datasets labeled A1, A2, A3, A4, B1, B2, C1, C2, and C3. The list of files in each of these datasets is given in separate text files located under `list` folder in the samples package[1]. Note that the datasets are subsets of each other (i.e., $C1 \subset C2 \subset C3 \subset B1 \ldots A3 \subset A4$).

All the datasets are available under `file` folder. To create a dataset (e.g., A1), run the `copy.py` script as the following:

```
python copy.py DATASET_LABEL
```

Where DATASET_LABEL is one of the dataset labels. This scrip creates a folder named after the dataset label, and copies all the files belonging to that dataset from `file` folder to the newly created folder.

[1] The package containing these data, can be downloaded from the following page: https://genometric.github.io/Di4/benchmark

3. Environment setup

Di4, Di3, BEDTools and BEDOPS run in different environment with different requirements. Please consider the following table.

	OS requirement	Required libraries and packages
Di3 and Di4	Windows 10	.NET framework 4.7.1
BEDTools	Linux and Mac	http://bedtools.readthedocs.io/en/latest/content/installation.html
BEDOPS	Linux and Mac	http://bedops.readthedocs.io/en/latest/content/installation.html

4. How to run Di4/Di3?

Di3 and Di4 are accessible from a single package available from Di4's releases page at the following link:

https://github.com/Genometric/Di4/releases

Albeit file names refer to Di4 only. Additionally, the commands, their syntax, and their interface is unified; therefore, one can switch between Di3 and Di4 with two configuration keys (explained in section 4.1.5) and run them similarly. The descriptions in this section mainly refer to Di4, but all descriptions apply to Di3 as well. In the following, we explain how to setup Di4, and then we explain how to run these tools.

Note

We captured a demo of running Di4 in its basic setup, and it is available from the following link on YouTube:

https://youtu.be/Tnt5kFHUqZI

The demo shows how to run Di4 to index files in the dataset B2 and execute the **Map** function using the given reference file.

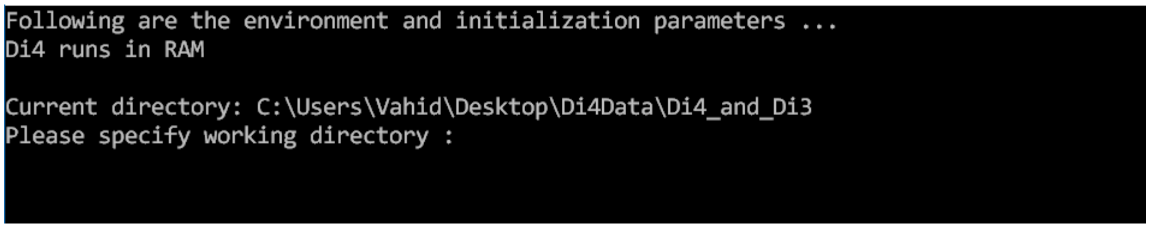
4.1. Configuration parameters

Di4 requires some configuration parameters to be set. First, we explain how to set the parameters, and then we explain each of the parameters and their role.

4.1.1. How to set configuration parameters?

There are two methods to set configuration parameters:

1. **Through command line interface (CLI)**: when running Di4 for the first time (see section 4.2), Di4 prompts for required configuration parameters and saves them for future use. For instance, at its first execution Di4 asks for **`WorkingDirectory`** as shown in the following figure:



```
Following are the environment and initialization parameters ...
Di4 runs in RAM

Current directory: C:\Users\Vahid\Desktop\Di4Data\Di4_and_Di3
Please specify working directory :
```

2. **Manually modify Di4 configuration file, `Di4BCLI.exe.Config`**: this file is located under the folder extracted from the downloaded Di4 release (https://github.com/Genometric/Di4/releases) and can be opened using any text editor (e.g., Notepad, or Notepad++). In its basic setup, its content is as the following:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="Memory" value="RAM" />
    <add key="Parser__ChrColumn" value="0" />
    <add key="Parser__LeftEndColumn" value="1" />
    <add key="Parser__RightEndColumn" value="2" />
    <add key="Parser__NameColumn" value="3" />
    <add key="Parser__ValueColumn" value="4" />
  </appSettings>
  <runtime>
    <gcAllowVeryLargeObjects enabled="true"/>
  </runtime>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.1"/>
  </startup>
</configuration>
```

Some basic parameters can also be set via CLI, but some others can be set via configuration file only. Some basic configuration parameters are discussed in the following. Note that all the following settings should be nested under **`appSettings`** configuration key, as shown in the above example.

4.1.2. Run in RAM or HDD?

One can choose whether to index files in RAM (execute in-memory) or persist on hard disk. When persisted on hard disk, the indexed data can be re-used (and updated) in future execution. This configuration can be set via configuration file only. To set Di4 to execute in memory, set the following in the configuration file:

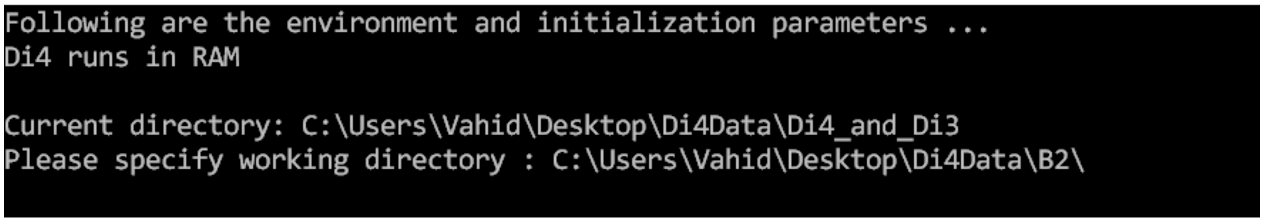
```
<add key="Memory" value="RAM" />
```

Moreover, to set Di4 execute from hard disk, set the following:

```
<add key="Memory" value="HDD" />
```

4.1.3. Working directory

Working directory is a folder where Di4 reads data to be indexed, and persists (if HDD is chosen, as aforementioned) index data structures. When testing Di4 on each of the datasets, set working directory to the dataset path. For instance, the following figure shows how to set working directory in CLI:



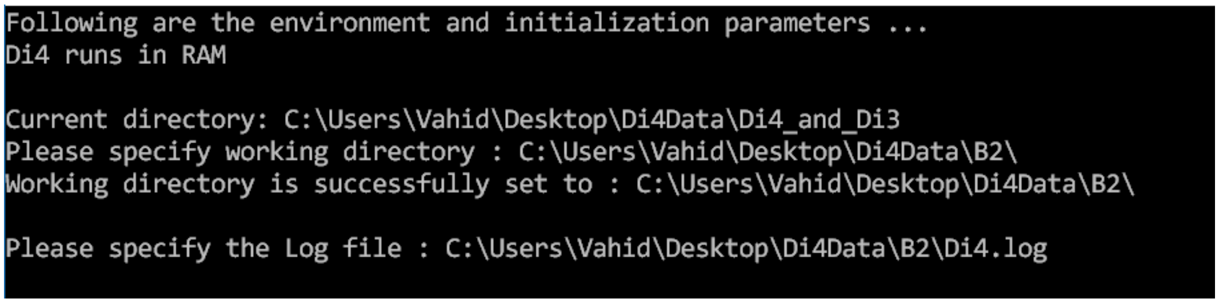
```
Following are the environment and initialization parameters ...
Di4 runs in RAM

Current directory: C:\Users\Vahid\Desktop\Di4Data\Di4_and_Di3
Please specify working directory : C:\Users\Vahid\Desktop\Di4Data\B2\
```

Alternatively, use the following key in configuration file.

```
<add key="WorkingDirectory" value="C:\Users\Vahid\Desktop\Di4Data\B2\" />
```

4.1.4. Log file

In addition to CLI interactions, Di4 logs in a text file every command it receives and every message it throws because of executing commands. One may set the log file as the following via CLI:



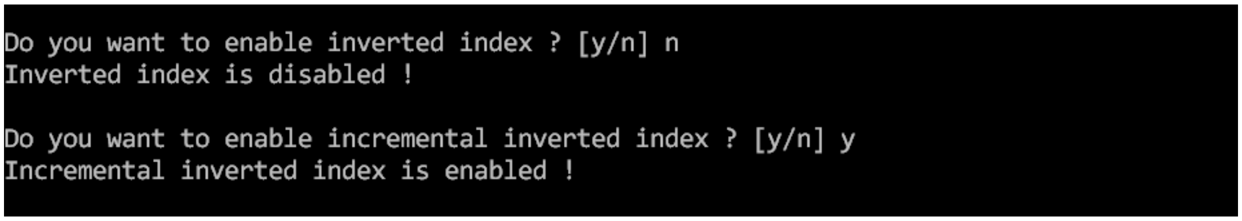
```
Following are the environment and initialization parameters ...
Di4 runs in RAM

Current directory: C:\Users\Vahid\Desktop\Di4Data\Di4_and_Di3
Please specify working directory : C:\Users\Vahid\Desktop\Di4Data\B2\
Working directory is successfully set to : C:\Users\Vahid\Desktop\Di4Data\B2\

Please specify the Log file : C:\Users\Vahid\Desktop\Di4Data\B2\Di4.log
```

Alternatively, using the following key in configuration file:

```
<add key="LogFile" value="C:\Users\Vahid\Desktop\Di4Data\B2\Di4.log" />
```

4.1.5. Enable/disable Di4/Di3

Di4 and Di3 can be enabled and/or disabled using two configuration keys, which can be set via both CLI and the configuration file. The following figure shows how they are set via CLI.



```
Do you want to enable inverted index ? [y/n] n
Inverted index is disabled !

Do you want to enable incremental inverted index ? [y/n] y
Incremental inverted index is enabled !
```

Inverted index refers to Di3, and **incremental inverted index** refers to Di4. To enable/disable Di4/Di3 via configuration file, the **`EnableInvertedIndex`** and **`EnableIncrementalIndex`** keys should be set, which refer to Di3 and Di4 respectively. Therefore, to run Di4 set the keys as the following:

```
<add key="EnableInvertedIndex" value="n" />
<add key="EnableIncrementalIndex" value="y" />
```

Alternatively, to run Di3, set the keys as the following:

```
<add key="EnableInvertedIndex" value="y" />
<add key="EnableIncrementalIndex" value="n" />
```

4.1.6. Cache size

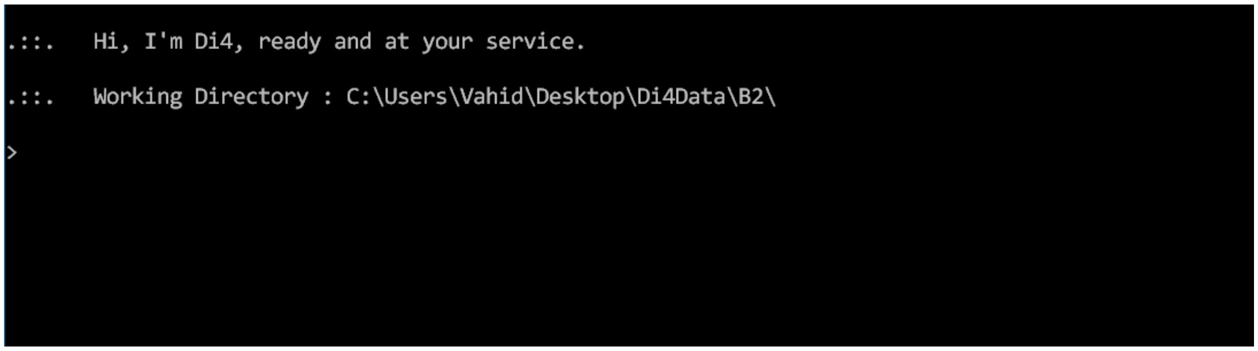
When executed in persisted mode (i.e., index data are stored on hard disk), Di4 can cache a number of recently visited snapshots in memory. The minimum and maximum number of snapshots to be cached can be specified using the following configuration keys.

```
<add key="MinBInCache" value="10" />
<add key="MaxBInCache" value="100" />
```

Determining an optimal cache size is a common challenge and applies here as well. The cache size is a trade-off between performance and memory consumption. If set too low, Di4 mainly relies on de-serializing snapshots from hard disk; and if set too high, depending on the computer performance, it may penalize performance. Additionally, whether the cached snapshots will be used executing a query or not, depends on the query. For instance, when Di4 executes a query that is targeting different parts of the genome at each step, the snapshots required to answer the query may not be cached previously; therefore, Di4's query on the cached data mainly hits no snapshot. This may degrade performance, as Di4 needs to check first the cached snapshots (which in this scenario may mainly hit no snapshot) before de-serializing required snapshots from hard disk. Accordingly, cache size is a factor of computer performance, memory size, input data, and commonly executed queries. Therefore, we leave it to the user to decide what could be "optimal" parameters for their setting.

4.2. Run Di4

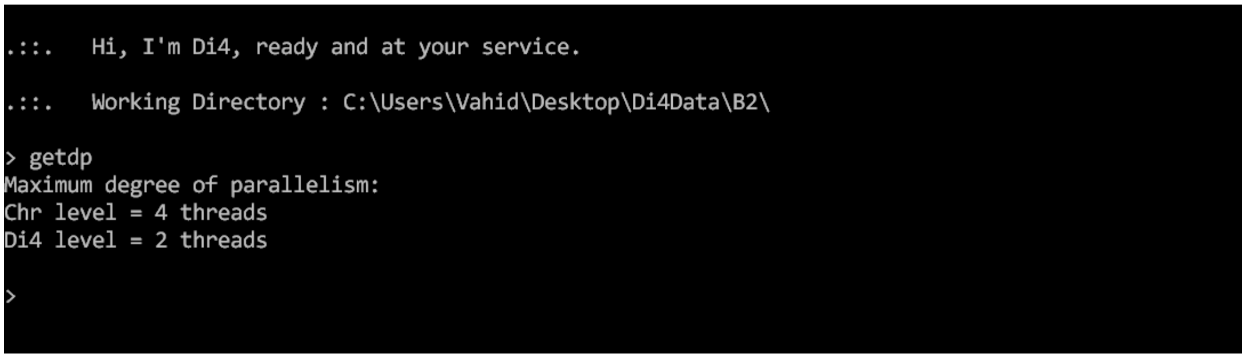
A simplest approach to run Di4 is by double-clicking on the `Di4BCLI.exe` file (extracted from the zip file downloaded from Di4 release page https://github.com/Genometric/Di4/releases). After setting/showing the configurations (see section 4.1), it shows the following screen which is Di4's command prompt.



```
.::.   Hi, I'm Di4, ready and at your service.

.::.   Working Directory : C:\Users\Vahid\Desktop\Di4Data\B2\

>
```

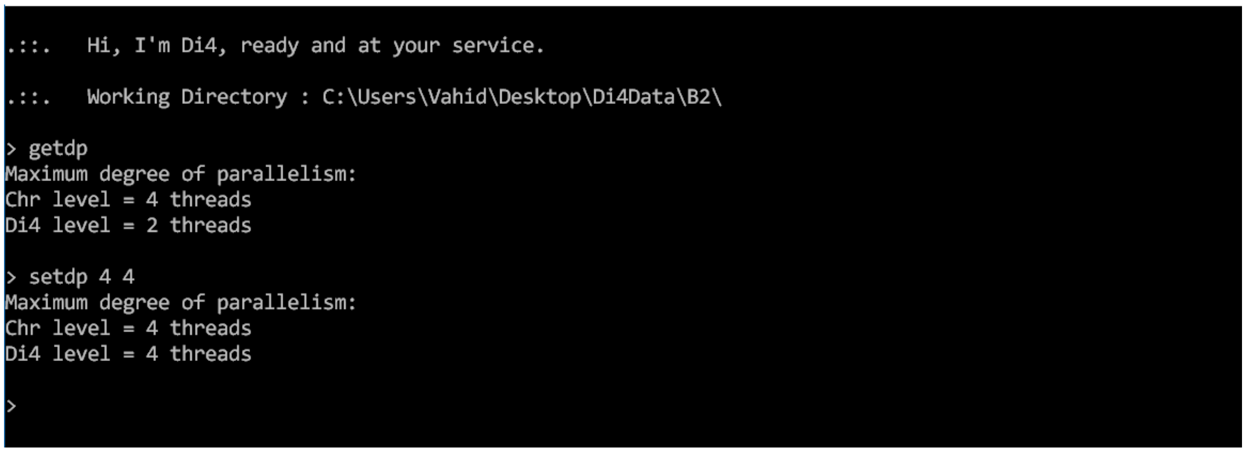
4.3. Set degree of parallelism

Run the command getdp in the Di4 command prompt (write the command and hit Enter); it displays a message as the following:



```
.::.   Hi, I'm Di4, ready and at your service.

.::.   Working Directory : C:\Users\Vahid\Desktop\Di4Data\B2\

> getdp
Maximum degree of parallelism:
Chr level = 4 threads
Di4 level = 2 threads

>
```

This command returns the degree of parallelism at which Di4 is running. Its default value is automatically determined based on the number of cores and threads of the host computer's processor. ($4 \times 2 = 16$ is the number of threads this test machine can run optimally in parallel.) Di4 extensively benefits from parallelization and goes under extensive (partial) locking mechanisms; therefore, its performance depends on the degree of parallelism. One may change the degree of parallelism by running setdp command as the following:

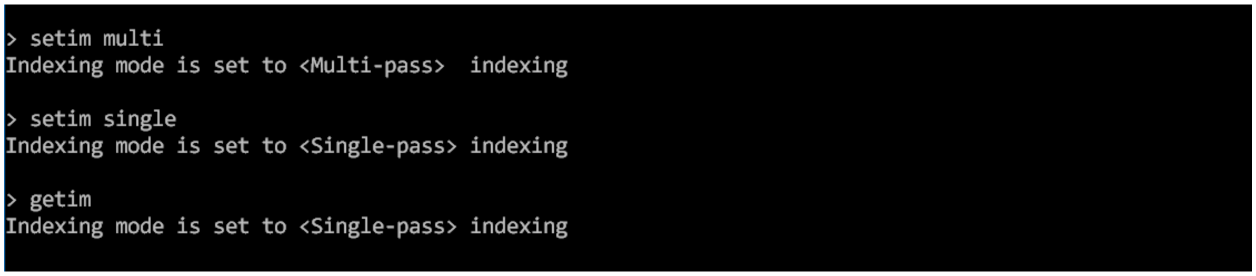


```
.::.   Hi, I'm Di4, ready and at your service.

.::.   Working Directory : C:\Users\Vahid\Desktop\Di4Data\B2\

> getdp
Maximum degree of parallelism:
Chr level = 4 threads
Di4 level = 2 threads

> setdp 4 4
Maximum degree of parallelism:
Chr level = 4 threads
Di4 level = 4 threads

>
```

Chr-level degree of parallelism specifies how many independent instances of Di4 run in parallel. Di4-level degree of parallelism indicates how many parallel threads read/write each Di4 instance. Accordingly, with Chr and Di4 level of parallelism set to four and four respectively, 16 (=4*4) threads are executing a query. Accordingly, the host machine's CPU and hard disk I/O (if index data are persisted on hard disk) should have enough throughput to execute all these threads optimally. Therefore, we recommend considering a host machine's specs for setting the degree of parallelism. Also, note that Di4 running in auto-determined degree of parallelism could perform at a reasonable throughput in most scenarios.

4.4. Set indexing mode

Di4 can index data in two indexing modes: single and double pass indexing. Use `setim single` or `setim multi` to set indexing mode to single and multi-pass modes respectively. One can also get the current setting for indexing mode using `getim` command. See the following console output as an example.

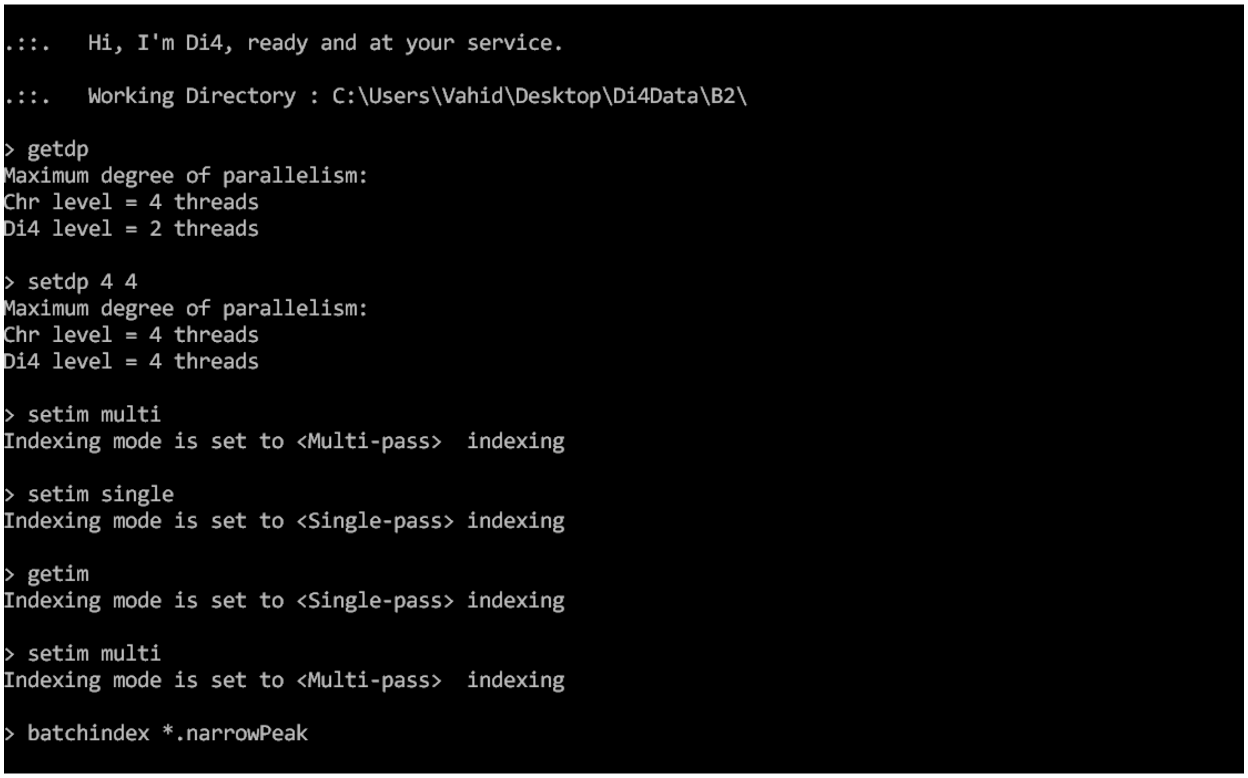


```
> setim multi
Indexing mode is set to <Multi-pass>  indexing

> setim single
Indexing mode is set to <Single-pass> indexing

> getim
Indexing mode is set to <Single-pass> indexing
```

4.5. Batch index

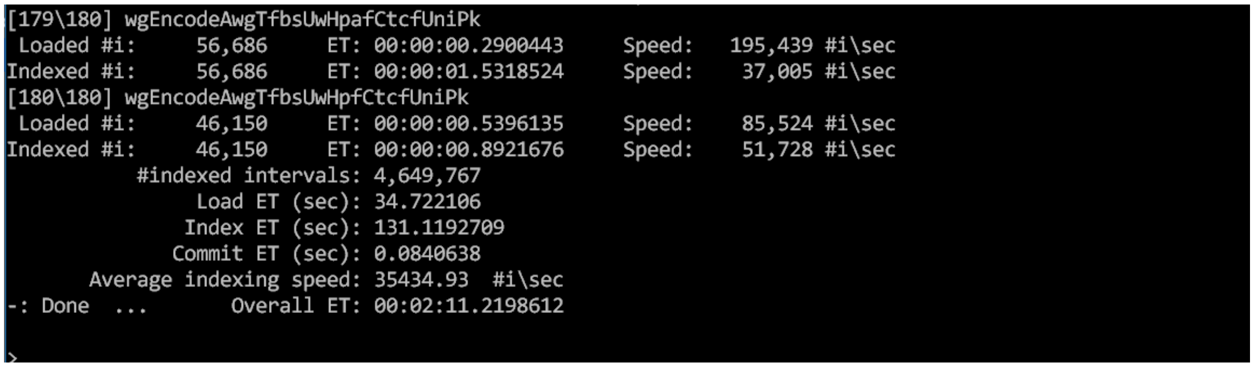
Di4 indexes data in batch; user specifies the files to be indexed, then Di4BCLI and Di4B parses the files one at a time, and indexes them in Di4. To index the data in the provided datasets, one may use a command similar to the following (note, we recommend setting `setim multi` prior to batch index):

```
batchindex *.narrowPeak
```



```
.::.   Hi, I'm Di4, ready and at your service.

.::.   Working Directory : C:\Users\Vahid\Desktop\Di4Data\B2\

> getdp
Maximum degree of parallelism:
Chr level = 4 threads
Di4 level = 2 threads

> setdp 4 4
Maximum degree of parallelism:
Chr level = 4 threads
Di4 level = 4 threads

> setim multi
Indexing mode is set to <Multi-pass>  indexing

> setim single
Indexing mode is set to <Single-pass> indexing

> getim
Indexing mode is set to <Single-pass> indexing

> setim multi
Indexing mode is set to <Multi-pass>  indexing

> batchindex *.narrowPeak
```

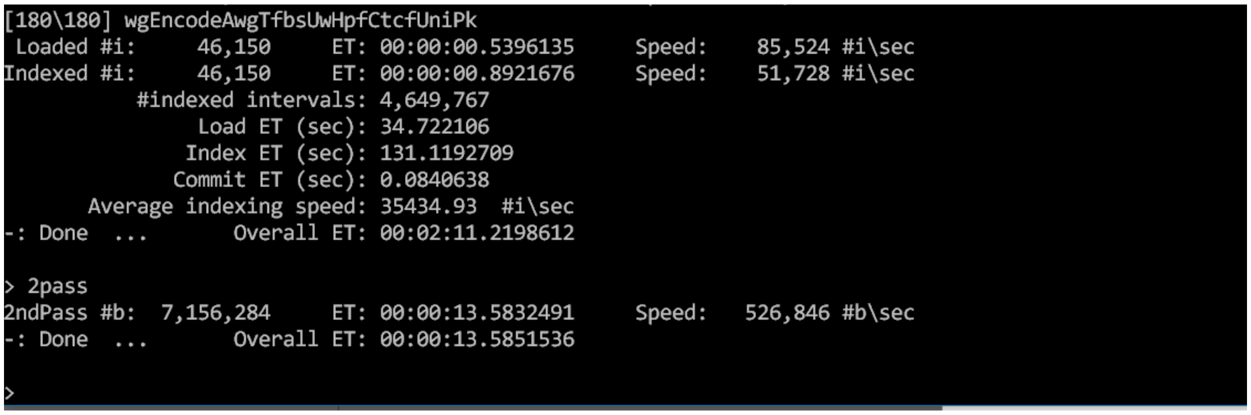
This command parses and indexes all the narrowpeaks of a dataset (the dataset is specified via the `WorkingDirectory` configuration); and it produces the following output when the first-pass (if multi-pass indexing was selected) of indexing is finished.



```
[179\180] wgEncodeAwgTfbsUwHpafCtcfUniPk
 Loaded #i:     56,686     ET: 00:00:00.2900443     Speed:   195,439 #i\sec
Indexed #i:     56,686     ET: 00:00:01.5318524     Speed:    37,005 #i\sec
[180\180] wgEncodeAwgTfbsUwHpfCtcfUniPk
 Loaded #i:     46,150     ET: 00:00:00.5396135     Speed:    85,524 #i\sec
Indexed #i:     46,150     ET: 00:00:00.8921676     Speed:    51,728 #i\sec
           #indexed intervals: 4,649,767
                Load ET (sec): 34.722106
               Index ET (sec): 131.1192709
              Commit ET (sec): 0.0840638
       Average indexing speed: 35434.93  #i\sec
-: Done  ...       Overall ET: 00:02:11.2198612

>
```

4.6. Second-pass index

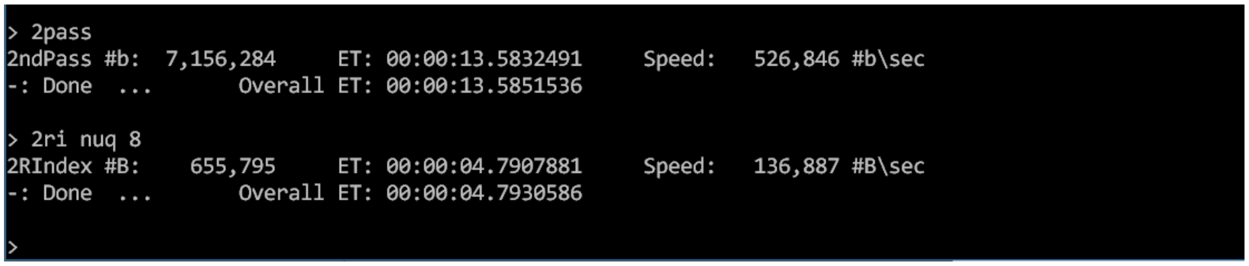
Run 2pass command to execute the second pass of double-pass indexing algorithm. Note, this command is not required if indexing mode was set to single-pass indexing.



```
[180\180] wgEncodeAwgTfbsUwHpfCtcfUniPk
 Loaded #i:     46,150     ET: 00:00:00.5396135     Speed:    85,524 #i\sec
Indexed #i:     46,150     ET: 00:00:00.8921676     Speed:    51,728 #i\sec
           #indexed intervals: 4,649,767
                Load ET (sec): 34.722106
               Index ET (sec): 131.1192709
              Commit ET (sec): 0.0840638
       Average indexing speed: 35434.93  #i\sec
-: Done  ...       Overall ET: 00:02:11.2198612

> 2pass
2ndPass #b:  7,156,284     ET: 00:00:13.5832491     Speed:   526,846 #b\sec
-: Done  ...       Overall ET: 00:00:13.5851536

>
```

4.7. Second resolution indexing

Run 2ri nuq 8 command to execute second-resolution indexing using pdf-optimized scalar quantization with 8 quantization levels. Other quantization methods are zt and uq for zero-thresholding and uniform quantization, where zero-thresholding does not take quantization level parameter.



```
> 2pass
2ndPass #b:  7,156,284     ET: 00:00:13.5832491     Speed:   526,846 #b\sec
-: Done  ...       Overall ET: 00:00:13.5851536

> 2ri nuq 8
2RIndex #B:    655,795     ET: 00:00:04.7907881     Speed:   136,887 #B\sec
-: Done  ...       Overall ET: 00:00:04.7930586

>
```

Some example of calling second-resolution indexing:

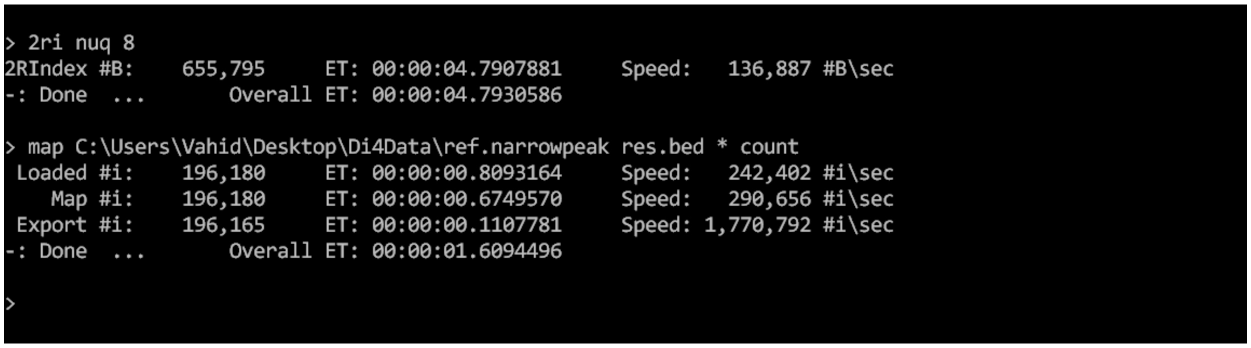
```
2ri zt

2ri uq 8

2ri nuq 16
```

4.8. Execute Map function

The map function takes a reference file, an output filename (will be saved to working directory by default), the chromosome strand on which the operation should be applied, and an aggregation function. For instance:

```
map C:\Users\Vahid\Desktop\Di4Data\ref.narrowpeak res.bed * count
```

Where * means “un-stranded” as the provided input data do not define strand. Count is an aggregation function, which counts the number of intervals overlapping a reference interval. Because of executing this command, Di4BCLI shows the following message.



```
> 2ri nuq 8
2RIndex #B:    655,795     ET: 00:00:04.7907881     Speed:   136,887 #B\sec
-: Done  ...        Overall ET: 00:00:04.7930586

> map C:\Users\Vahid\Desktop\Di4Data\ref.narrowpeak res.bed * count
 Loaded #i:    196,180     ET: 00:00:00.8093164     Speed:   242,402 #i\sec
    Map #i:    196,180     ET: 00:00:00.6749570     Speed:   290,656 #i\sec
 Export #i:    196,165     ET: 00:00:00.1107781     Speed: 1,770,792 #i\sec
-: Done  ...        Overall ET: 00:00:01.6094496

>
```

Regarding the runtime: Di4's runtime is the runtime of the Map function, which is 00:00:00:6749570 in this run. The load and export times are Di4B and Di4BCLI runtime for loading the reference sample, calling independent instances of Di4 instances (chr-level degree of parallelism), and saving the results.

Note

Consider the following points when running Di4 on a given dataset:

1. Reset the configuration file;
2. Make sure the working directory points to the dataset which you want to index in Di4;
3. If running in persisted mode, make sure the previous index files either are deleted or are located in different a working directory, if you don't intent to update the existing index.

5. How to run BEDTools and BEDOPS?

In order to simplify executing BEDTools and BEDOPS, we prepared a python script, which should run all the necessary steps for executing BEDTools and BEDOPS. The script file name is **`run.py`** and is available from both the demo package and https://github.com/Genometric/Di4/blob/master/scrips/run.py.

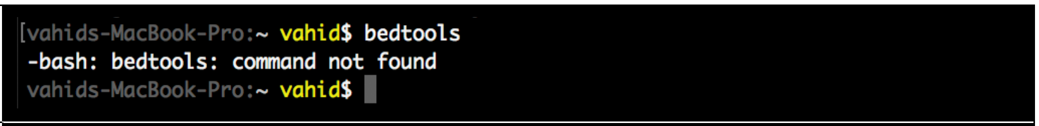
Please note, BEDTools and BEDOPS run on Linux or MacOS only (BEDOPS has a Cygwin-based execution for Windows users, but we have not tested against this feature). Hence, this script is executable in Linux or MacOS only. Please follow the following steps to run the scripts:

1. Install
   a. BEDTools: http://bedtools.readthedocs.io/en/latest/content/installation.html#
   b. BEDOPS: http://bedops.readthedocs.io/en/latest/content/installation.html
2. Make sure the path to projects are registered in user shell profile. You may test it by executing the following commands.

```
bedtools
bedmap
```

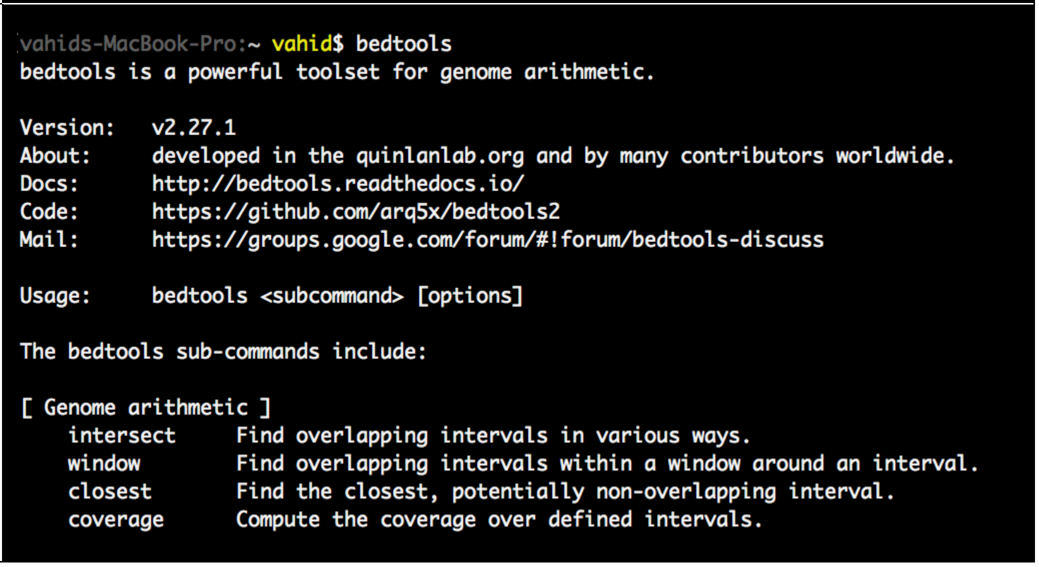
If the path to the projects are defined in shell profile, you may see the documentation of arguments of the projects; otherwise, you may see errors; see following screenshots:

If path is NOT defined



```
[vahids-MacBook-Pro:~ vahid$ bedtools
 -bash: bedtools: command not found
 vahids-MacBook-Pro:~ vahid$
```

If path is defined



```
vahids-MacBook-Pro:~ vahid$ bedtools
bedtools is a powerful toolset for genome arithmetic.

Version:   v2.27.1
About:     developed in the quinlanlab.org and by many contributors worldwide.
Docs:      http://bedtools.readthedocs.io/
Code:      https://github.com/arq5x/bedtools2
Mail:      https://groups.google.com/forum/#!forum/bedtools-discuss

Usage:     bedtools <subcommand> [options]

The bedtools sub-commands include:

[ Genome arithmetic ]
    intersect     Find overlapping intervals in various ways.
    window        Find overlapping intervals within a window around an interval.
    closest       Find the closest, potentially non-overlapping interval.
    coverage      Compute the coverage over defined intervals.
```

To register a path in user bash profile you may follow these guides:

- macOS: https://coolestguidesontheplanet.com/add-shell-path-osx/
- linux: https://stackoverflow.com/a/14638025/947889
  or https://unix.stackexchange.com/a/26059

3. Run the scripts from terminal such as the following:

```
python run.py TOOL_NAME DATASET [--on-the-fly]
```

where TOOL_NAME should be replaced by --bedtools or --bedops, and DATASET should be replaced by a dataset name which are a1 , a2 , a3 , a4 , b1 , b2 , c1 , c2 , and c3. The --on-the-fly key will set the script to include pre-processing time in the query time.