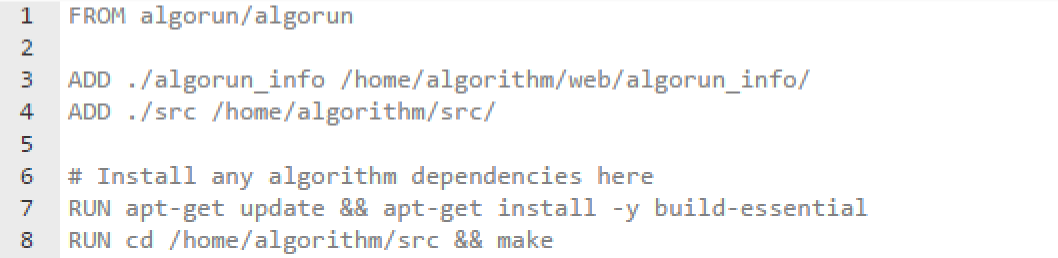
# 1 Packaging Bowtie Software with AlgoRun

Bowtie is an ultra-fast memory-efficient short read aligner[1]. The source code is written in C++ and is available under the Artistic License. Download it from http://sourceforge.net/projects/bowtie-bio/files/bowtie/1.1.2/

Unzip the downloaded file. It results in a folder containing all the source code of Bowtie.

STEP 1: Add all Bowtie source files inside the `src` folder.

STEP 2: Install C++ dependencies in Dockerfile. In addition, build the source code using `make` command. Below is how the Dockerfile of Bowtie looks like.



```
FROM algorun/algorun

ADD ./algorun_info /home/algorithm/web/algorun_info/
ADD ./src /home/algorithm/src/

# Install any algorithm dependencies here
RUN apt-get update && apt-get install -y build-essential
RUN cd /home/algorithm/src && make
```

*Source code: Dockerfile of Bowtie software*

***Hints:***

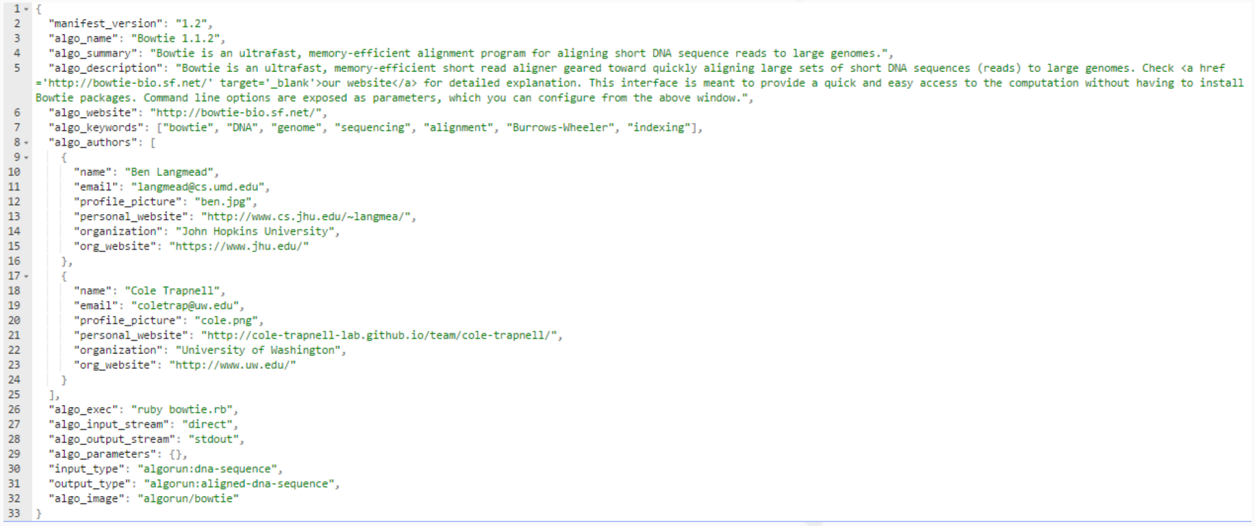
*1. Dockerfile syntax requires to precede all commands with* RUN *keyword.*

*2. To ensure successful installation, always use apt-get update before installing packages and use -y option in the install command.*

*3. Change to* `/home/algorithm/src` *directory before running any command that operates on the source files inside* `src` *folder.*

STEP 3: `manifest.json` file is required to describe the computational algorithm. Comments in the file will guide you to fill the correct values. Below is how the manifest of Bowtie looks like.

[1] For more information: visit Bowtie website: http://bowtie-bio.sourceforge.net/index.shtml



```
{
  "manifest_version": "1.2",
  "algo_name": "Bowtie 1.1.2",
  "algo_summary": "Bowtie is an ultrafast, memory-efficient alignment program for aligning short DNA sequence reads to large genomes.",
  "algo_description": "Bowtie is an ultrafast, memory-efficient short read aligner geared toward quickly aligning large sets of short DNA sequences (reads) to large genomes. Check <a href='http://bowtie-bio.sf.net/' target='_blank'>our website</a> for detailed explanation. This interface is meant to provide a quick and easy access to the computation without having to install Bowtie packages. Command line options are exposed as parameters, which you can configure from the above window.",
  "algo_website": "http://bowtie-bio.sf.net/",
  "algo_keywords": ["bowtie", "DNA", "genome", "sequencing", "alignment", "Burrows-Wheeler", "indexing"],
  "algo_authors": [
    {
      "name": "Ben Langmead",
      "email": "langmead@cs.umd.edu",
      "profile_picture": "ben.jpg",
      "personal_website": "http://www.cs.jhu.edu/~langmea/",
      "organization": "John Hopkins University",
      "org_website": "https://www.jhu.edu/"
    },
    {
      "name": "Cole Trapnell",
      "email": "coletrap@uw.edu",
      "profile_picture": "cole.png",
      "personal_website": "http://cole-trapnell-lab.github.io/team/cole-trapnell/",
      "organization": "University of Washington",
      "org_website": "http://www.uw.edu/"
    }
  ],
  "algo_exec": "ruby bowtie.rb",
  "algo_input_stream": "direct",
  "algo_output_stream": "stdout",
  "algo_parameters": {},
  "input_type": "algorun:dna-sequence",
  "output_type": "algorun:aligned-dna-sequence",
  "algo_image": "algorun/bowtie"
}
```

*Source Code: **manifest.json** of Bowtie software (comments-skimmed)*

STEP 4:

- `input_example.txt` file includes a sample input data for users to quickly try the algorithm. Enter `ATGCATCATGCGCCAT` as an example.
- `output_example.txt` file includes a sample of the expected output for the same input. It makes it easier for users to expect the results. The above input produces the following:

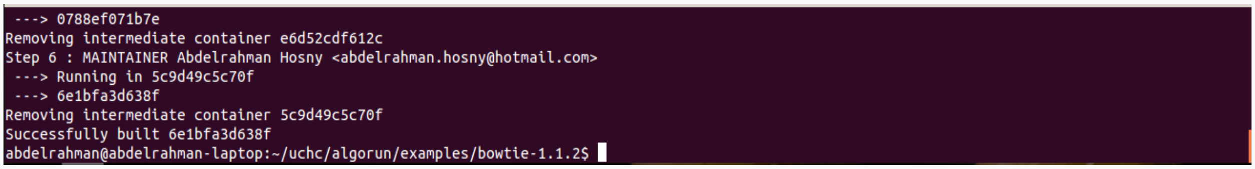
```
0     -     gi|110640213|ref|NC_008253.1| 148810      ATGGCGCATGATGCAT
      IIIIIIIIIIIIIIII  0     10:A>G,13:C>G
```

NOTES

- Bowtie source code comes with `e_coli` index packaged by default. So, use it in the `algo_exec`. If you included other indexes, it’s ok to use them as well.
- Use `direct` in `algo_input_stream` to accept input directly from the command line. Bowtie has other options to read the input from a file. However, AlgoRun will automatically present an option to upload a file to the input area in the web interface.
- Use `stdout` in `algo_output_stream` to let AlgoRun get the result from the terminal. Bowtie has other options to write the output to a file. However, AlgoRun will automatically present an option to download the result to a file from the web interface.

STEP 5:

- From the directory where the Dockerfile exists, build Bowtie container using:
  `docker build -t bowtie .`
- You should see a success message as in the following picture.



```
 ---> 0788ef071b7e
Removing intermediate container e6d52cdf612c
Step 6 : MAINTAINER Abdelrahman Hosny <abdelrahman.hosny@hotmail.com>
 ---> Running in 5c9d49c5c70f
 ---> 6e1bfa3d638f
Removing intermediate container 5c9d49c5c70f
Successfully built 6e1bfa3d638f
abdelrahman@abdelrahman-laptop:~/uchc/algorun/examples/bowtie-1.1.2$
```

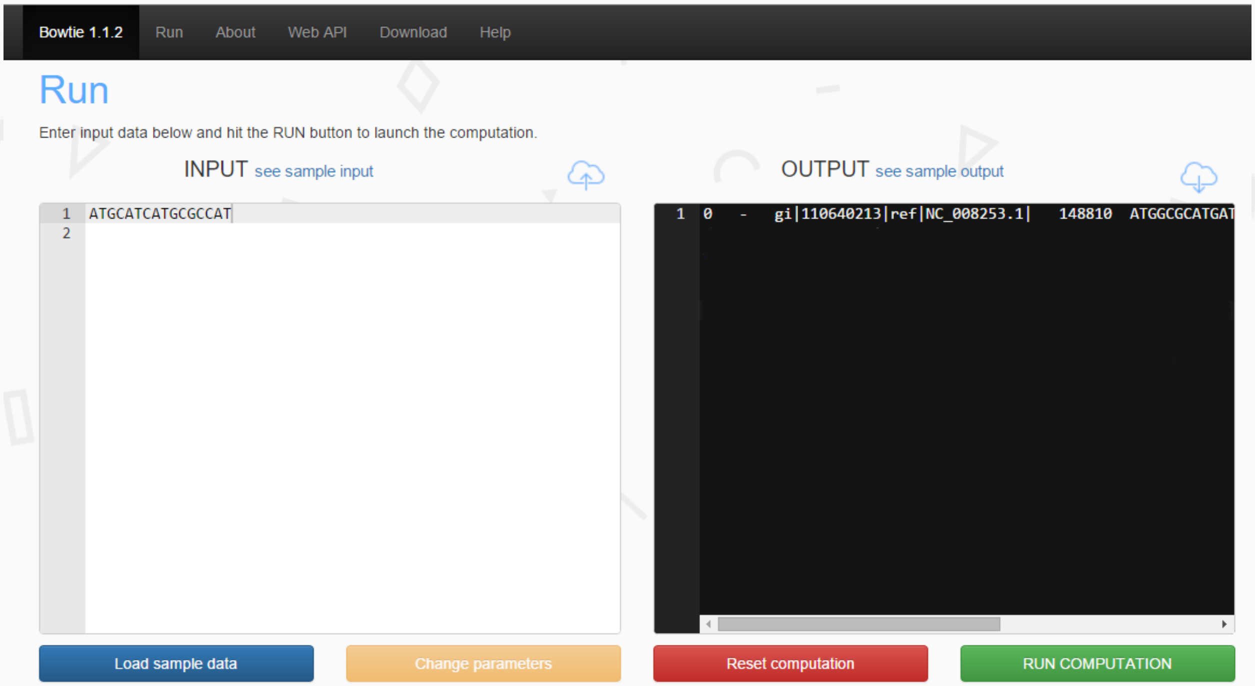
*Bowtie container build success message*

## 2 User Interface

Run Bowtie container using:
`docker run -p 31331:8765 bowtie`

Open the web browser and type http://localhost:31331

*Hint: You can use any available port other than 31331. Yet, you must bind it to 8765 port as it is the gateway to AlgoRun.*



```
ATGCATCATGCGCCAT
```

```
0	-	gi|110640213|ref|NC_008253.1|	148810	ATGGCGCATGAT
```

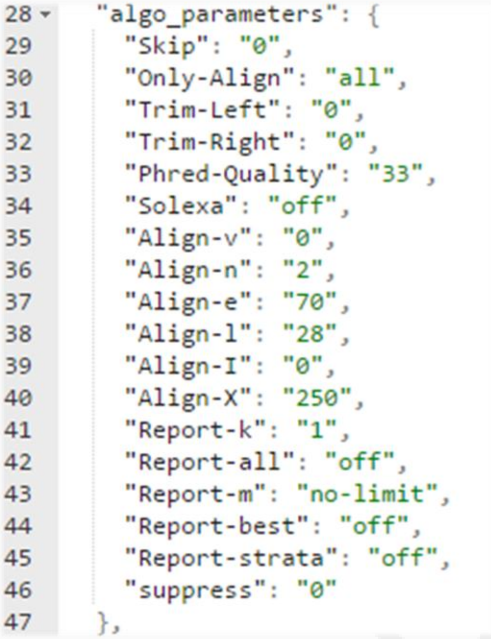
## 3 [OPTIONAL] Expose Command Line Options as Parameters

To give a computational algorithm flexibility, AlgoRun allows to expose parameters that can be easily changed from the web interface. These parameters will be available as environment variables in the source code.

The power of Bowtie as a very fast DNA sequences aligner comes from the available command line options. So, you can make use of AlgoRun parameters to expose these command line options. You have two options: either to manipulate the source code of Bowtie so that it reads options from environment variables (instead of command line) or to develop a wrapper around Bowtie main executable that will internally translate environment variables to command line options. To do so, follow the below steps:

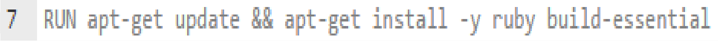
*The example here uses Ruby programming language to write the wrapper. You can use any other language and apply the same concepts.*

1. Specify parameters and their default values in the manifest file. The adjacent picture shows some parameters.
2. Read the input data[2]. The input data is passed as the first command line argument.
3. Read the environment variables (of the same names you specified in the manifest) and form the options string.
4. Call the executable file and to print the output to the standard output[3].



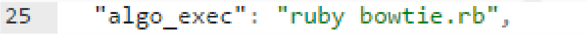
```
    "algo_parameters": {
      "Skip": "0",
      "Only-Align": "all",
      "Trim-Left": "0",
      "Trim-Right": "0",
      "Phred-Quality": "33",
      "Solexa": "off",
      "Align-v": "0",
      "Align-n": "2",
      "Align-e": "70",
      "Align-l": "28",
      "Align-I": "0",
      "Align-X": "250",
      "Report-k": "1",
      "Report-all": "off",
      "Report-m": "no-limit",
      "Report-best": "off",
      "Report-strata": "off",
      "suppress": "0"
    },
```

Modify the Dockerfile to install ruby dependency:



```
RUN apt-get update && apt-get install -y ruby build-essential
```

Modify `algo_exec` value in the manifest file to:

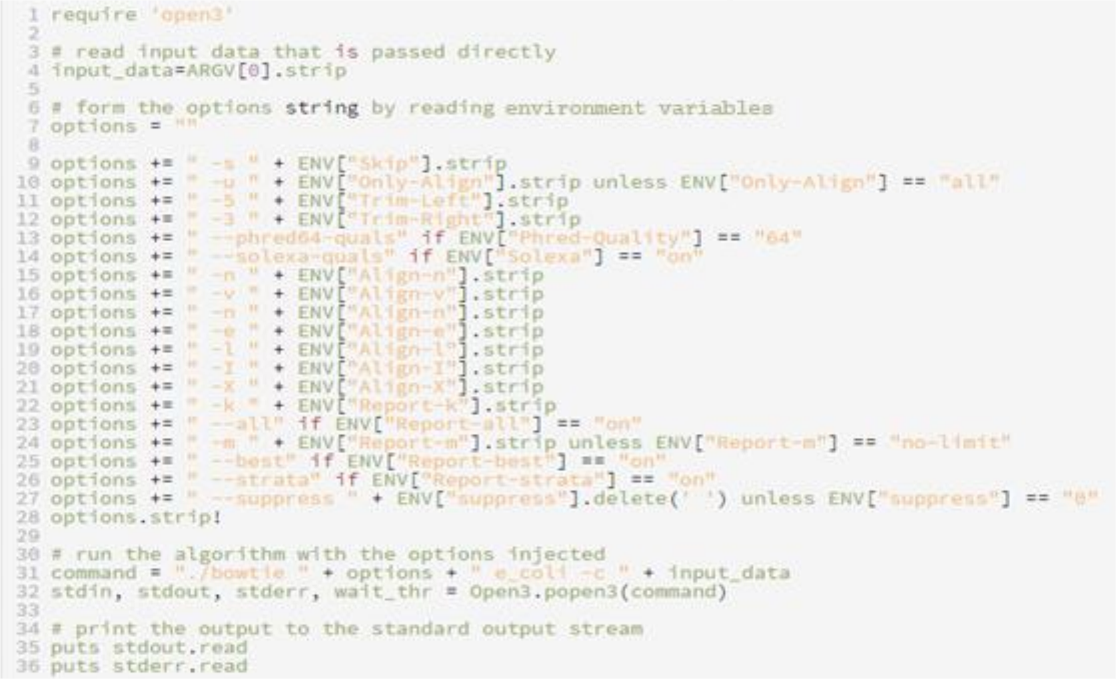


```
    "algo_exec": "ruby bowtie.rb",
```

Rebuild Bowtie container using: `docker build -t bowtie .`

[2] Remember that you can use “direct”, “file” or “stdin” alternatives to read the input. Whatever you choose, modify the manifest file accordingly.

[3] You can also write the output to a file and specify the file name in the manifest key “algo_output_stream”

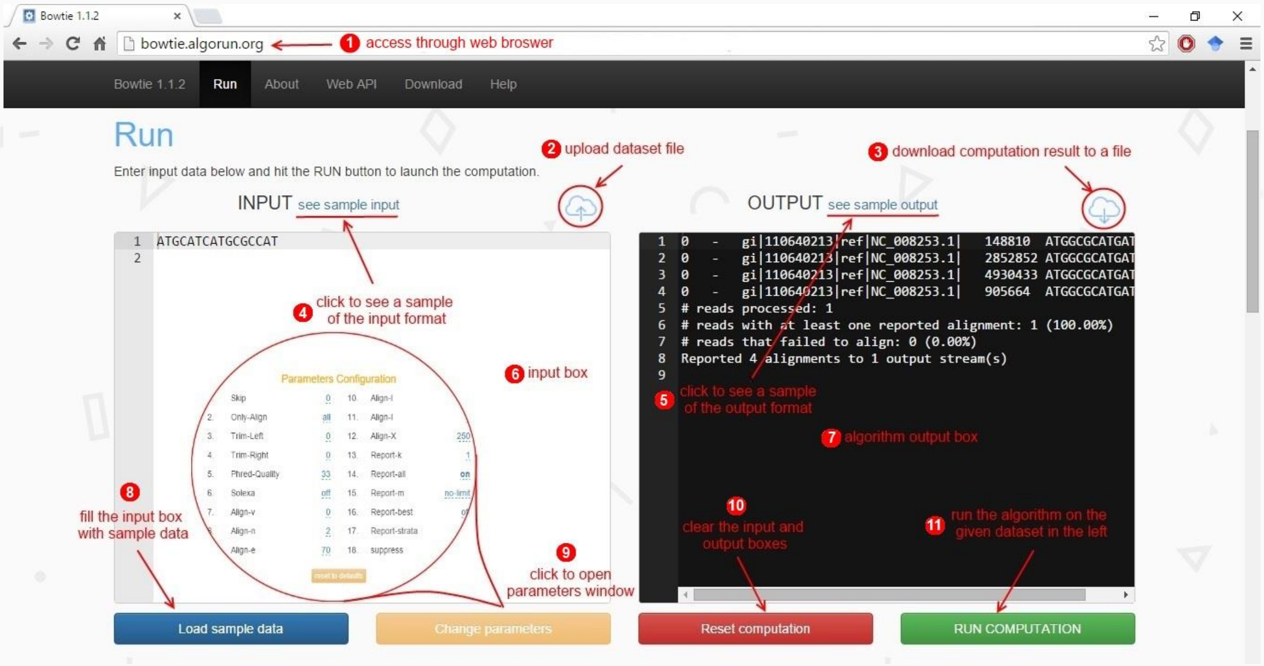


```
require 'open3'

# read input data that is passed directly
input_data=ARGV[0].strip

# form the options string by reading environment variables
options = ""

options += " -s " + ENV["Skip"].strip
options += " -u " + ENV["Only-Align"].strip unless ENV["Only-Align"] == "all"
options += " -5 " + ENV["Trim-Left"].strip
options += " -3 " + ENV["Trim-Right"].strip
options += " --phred64-quals" if ENV["Phred-Quality"] == "64"
options += " --solexa-quals" if ENV["Solexa"] == "on"
options += " -n " + ENV["Align-n"].strip
options += " -v " + ENV["Align-v"].strip
options += " -n " + ENV["Align-n"].strip
options += " -e " + ENV["Align-e"].strip
options += " -l " + ENV["Align-l"].strip
options += " -I " + ENV["Align-I"].strip
options += " -X " + ENV["Align-X"].strip
options += " -k " + ENV["Report-k"].strip
options += " --all" if ENV["Report-all"] == "on"
options += " -m " + ENV["Report-m"].strip unless ENV["Report-m"] == "no-limit"
options += " --best" if ENV["Report-best"] == "on"
options += " --strata" if ENV["Report-strata"] == "on"
options += " --suppress " + ENV["suppress"].delete(' ') unless ENV["suppress"] == "0"
options.strip!

# run the algorithm with the options injected
command = "./bowtie " + options + " e_coli -c " + input_data
stdin, stdout, stderr, wait_thr = Open3.popen3(command)

# print the output to the standard output stream
puts stdout.read
puts stderr.read
```

*Source Code: **bowtie.rb** wrapper code*

At this point, options available from Bowtie can be changed by clicking on "Change Parameters" button from the web interface. Visit http://bowtie.algorun.org for the final version of Bowtie running inside AlgoRun standard container.

Find the complete example on AlgoRun GitHub repository (https://github.com/algorun/algorun).



Bowtie 1.1.2
bowtie.algorun.org
1 access through web broswer
Bowtie 1.1.2
Run
About
Web API
Download
Help
Run
Enter input data below and hit the RUN button to launch the computation.
2 upload dataset file
3 download computation result to a file
INPUT see sample input
OUTPUT see sample output
4 click to see a sample of the input format
5 click to see a sample of the output format
6 input box
7 algorithm output box
8 fill the input box with sample data
9 click to open parameters window
10 clear the input and output boxes
11 run the algorithm on the given dataset in the left
Parameters Configuration
reset to defaults
Load sample data
Change parameters
Reset computation
RUN COMPUTATION

```
ATGCATCATGCGCCAT
```

| | | | | | |
|---|---|---|---|---|---|
| | Skip | 0 | 10. | Align-l | |
| 2. | Only-Align | all | 11. | Align-I | |
| 3. | Trim-Left | 0 | 12. | Align-X | 250 |
| 4. | Trim-Right | 0 | 13. | Report-k | 1 |
| 5. | Phred-Quality | 33 | 14. | Report-all | on |
| 6. | Solexa | off | 15. | Report-m | no-limit |
| 7. | Align-v | 0 | 16. | Report-best | of |
| 8 | Align-n | 2 | 17. | Report-strata | |
| | Align-e | 70 | 18. | suppress | |

```
0   -   gi|110640213|ref|NC_008253.1|   148810  ATGGCGCATGAT
0   -   gi|110640213|ref|NC_008253.1|   2852852 ATGGCGCATGAT
0   -   gi|110640213|ref|NC_008253.1|   4930433 ATGGCGCATGAT
0   -   gi|110640213|ref|NC_008253.1|   905664  ATGGCGCATGAT
# reads processed: 1
# reads with at least one reported alignment: 1 (100.00%)
# reads that failed to align: 0 (0.00%)
Reported 4 alignments to 1 output stream(s)
```

*Bowtie web interface*

## 4 Examples (Command Line Options vs. Parameters)

**4.1. Example Link:** http://bowtie-bio.sourceforge.net/manual.shtml#example-1--a

Command line:

```
./bowtie -a -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
```

With AlgoRun: Change `Report-all` to on, `Align-v` to 2 and `suppress` to 1,5,6,7

**4.2. Example Link:** http://bowtie-bio.sourceforge.net/manual.shtml#example-2--k-3

Command line:

```
./bowtie -k 3 -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
```

With AlgoRun: Change `Report-k` to 3, `Align-v` to 2 and `suppress` to 1,5,6,7

**4.3. Example Link:** http://bowtie-bio.sourceforge.net/manual.shtml#example-3--k-6

Command line:

```
./bowtie -k 6 -v 2 e_coli --suppress 1,5,6,7 -c ATGCATCATGCGCCAT
```

With AlgoRun: Change `Report-k` to 6, `Align-v` to 2 and `suppress` to 1,5,6,7

**4.4. Example Link:** http://bowtie-bio.sourceforge.net/manual.shtml#example-9--a--m-3---best---strata

Command line:

```
./bowtie -a -m 3 --best --strata -v 2 e_coli --suppress 1,5,6,7 -c
ATGCATCATGCGCCAT
```

With AlgoRun: Change `Report-all` to on, `Report-m` to 3, `Report-best` to on, `Report-strata` to on, `Align-v` to 2 and `suppress` to 1,5,6,7.