Neptune

Version 1.0.0

Eric Marinier

November 16, 2015

Contents

1 Introduction

Neptune locates genomic signatures using an exact k-mer matching strategy while accommodating k-mer mismatches. The software identifies sequences that are sufficiently represented within inclusion targets and sufficiently absent from exclusion targets. The signature discovery process is accomplished using probabilistic models instead of heuristic strategies. These general-purpose signatures are agnostic of application-specific requirements, such as physical and chemical properties, and may require further investigation to determine their appropriateness.

2 Installation

Neptune achieves maximum parallelization by submitting jobs through a Python DRMAA binding to a DRMAA-compliant scheduler. This installation will require a substantial understanding of Unix if such a DRMAA-compliant scheduler additionally needs to be installed and configured.

2.1 Requirements

Neptune should be installed and run on a standard 64-bit Linux environment. The software will attempt to automatically install many dependencies. However, the following dependencies must be manually installed by the user:

- Python 2.7
- NumPy
- SciPy
- DRMAA-compliant scheduler
- Python DRMAA
- BLAST+
- pipsi

2.2 DRMAA

Neptune uses a Python DRMAA binding to schedule parallel jobs. The user will first need to install and configure a DRMAA-compliant scheduler and further install the Python DRMAA bindings in order to use Neptune. Neptune has been tested using SGE installed on a single machine with the following instructions:

`https://scidom.wordpress.com/2012/01/18/sge-on-single-pc/`

However, any DRMAA-compliant scheduler is expected to work. The instructions for installing and configuring such scheduling environments are beyond the scope of this resource. The information necessary for installing and configuring the Python DRMAA bindings is available at:

`https://github.com/pygridtools/drmaa-python`

2.3 Neptune

Neptune executes in a Python environment and therefore does not need to be explicitly compiled. Neptune is best installed using pipsi by running the following command:

```
pipsi install /path/to/neptune/
```

If the Neptune installation was successful, you should be able to run the following without error:

```
neptune --help
```

3 Parameters

A short help message may be viewed by running:

```
neptune --help
```

3.1 Required Parameters

Neptune requires the location of the inclusion, exclusion, and output directories. The remaining parameters will be estimated based on the input sequence or revert to default settings. The following is the minimum number of command line parameters required to run Neptune:

```
neptune
  --inclusion /path/to/inclusion/
  --exclusion /path/to/exclusion/
  --output /path/to/output/
```

The following is a description of the required command line parameters:

inclusion
--inclusion -i [**file**]
A list of inclusion targets in FASTA format. You may list multiple file or directory locations following the **--inclusion** parameter. Neptune will automatically include all root-level files within directories.

exclusion
--exclusion -e [**file**]
A list of exclusion targets in FASTA format. You may list multiple file or directory locations following the **--exclusion** parameter. Neptune will automatically include all root-level files within directories.

output
--output -o [**directory**]
The location of the output directory. If this directory exists, any files produced with existing names will be overwritten. If this directory does not exist, then it will be created.

3.2 Optional Parameters

The following is a description of optional command line parameters:

help
--help -h
Displays a help message to standard output.

***k*-mer**

--kmer -k [**int**]

The size of the k-mers. This must be a positive integer and should be large enough such that random intra-genome k-mer matches, within the largest genome, are unexpected. The size of k-mers cannot be smaller than the smallest sequence record.

parallelization

--parallelization -p [**int**]

The degree of parallelization of k-mer aggregation. This determines the number of files to which the k-mers for each target are written and the number of parallel instances of k-mer aggregation. The number of parallel instances is determined by 4^p, where p is the specified parallelization argument (--parallelization). This value must be a non-negative integer smaller than k. If the parameter is not specified, then $p = 0$ and there will be no parallel k-mer aggregation. This will likely require a much longer computation time.

reference

--reference -r [**file**]

A list of references from which to extract signatures. If this parameter is not specified, signatures will be extracted from **all** inclusion targets. You may list multiple file locations following the **--reference** parameter.

rate

--rate -q [**float**]

This is the probability (0.0, 1.0) that any two homologous bases are different from each other. This should incorporate mutation rates, sequencing error rates, and assembly error rates. The rate is used to calculate the maximum allowable gap size in a signature and the minimum expected number of exact k-mer matches in a signature. If this value is not specified, the rate is assumed to be 0.01.

GC-content

--gcContent -gc [**float**]

The expected GC-content of the environment. The GC-content is used to calculate the maximum allowable gap size in a signature and the minimum expected number of exact k-mer matches in a signature. If this value is not specified, it is calculated by observing the GC-content of each target during signature extraction. The value must be between (0.0, 1.0).

confidence

--confidence -c [**float**]

The statistical confidence of decision making in the software. The confidence affects the automatic calculation of both the maximum gap size and minimum number of inclusion hits. If this value is not specified, a default of 0.95 is used. The value must be between (0.0, 1.0).

minimum inclusion hits

--inhits -ih [**int**]

The minimum number of inclusion hits required to start and continue signature extraction. If this value is not specified, it will be automatically calculated using the number of inclusion targets, the GC-content, the rate, and the k-mer size. The calculation can be found in the *Mathematics* documentation. This value must be a positive integer.

minimum exclusion hits

--exhits -eh [int]

The minimum number of exclusion hits necessary to stop extraction of a signature. If this value is not specified, it is assumed to be 1. This value must be a positive integer.

maximum gap size

--gap -g [int]

The maximum allowable number of base positions shifted before seeing an exact k-mer match. If this value is not specified, it will be automatically calculated using the rate, GC-content, and the k-mer size. The calculation can be found in the *Mathematics* documentation. This value must be a positive integer.

minimum signature size

--size -s [int]

The minimum size for a signature. Signatures which are shorter than this length will not be reported. If this value is not specified, the minimum signature size will be four times the length of the k-mer size ($4k$). It is not recommended to locate signatures smaller than this size, unless application-specific. This value must be a positive integer.

filter length

--filterLength -fl [float]

The minimum percent length of a exclusion hit against a signature candidate required to filter out the candidate. This value is a percentage expressed as a floating point number (0.0, 1.0). If the any exclusion hit exceeds the percent length **and** percent identity of any candidate, the candidate is removed. If this value is not specified, it is assumed to be 0.5.

filter percent

--filterPercent -fp [float]

The minimum percent identity of an exclusion hit against a signature candidate required to filter out the candidate. The percent identity is calculated as identities divided by the alignment length. This value is a percentage expressed as a floating point number (0.0, 1.0). If the any exclusion hit exceeds the percent length **and** percent identity of any candidate, the candidate is removed. If this value is not specified, it is assumed to be 0.5.

FILTERING NOTE

In order for a candidate to be filtered, any exclusion hit needs to exceed **both** the percent length (--filterLength) and percent identity (--filterPercent) thresholds.

3.3 DRMAA-Specific Arguments

It may be necessary to specify job submission parameters that are required by your parallel computing environment. If you require DRMAA-specific command line arguments, they may be provided to Neptune using one of several arguments. The **--defaultSpecification** parameter will provide the DRM-specific arguments to all jobs which are created. The remaining command line arguments allow specific arguments to each type of job.

default specification

--defaultSpecification [string]
DRMAA-specific command line arguments for all jobs. These arguments must be provided as a quoted string. The default specification will be applied to all job types and overwritten when specified.

count specification

--countSpecification [string]
DRMAA-specific command line arguments for *k*-mer counting. These arguments must be provided as a quoted string. These arguments will overwrite the default specification, if specified, for this job type.

aggregate specification

--aggregateSpecification [string]
DRMAA-specific command line arguments for *k*-mer aggregation. These arguments must be provided as a quoted string. These arguments will overwrite the default specification, if specified, for this job type.

extract specification

--extractSpecification [string]
DRMAA-specific command line arguments for signature extraction. These arguments must be provided as a quoted string. These arguments will overwrite the default specification, if specified, for this job type.

database specification

--databaseSpecification [string]
DRMAA-specific command line arguments for database construction. These arguments must be provided as a quoted string. These arguments will overwrite the default specification, if specified, for this job type.

filter specification

--filterSpecification [string]
DRMAA-specific command line arguments for candidate signature filtering. These arguments must be provided as a quoted string. These arguments will overwrite the default specification, if specified, for this job type.

4 Examples

4.1 Basic

The following example will allow Neptune to automatically calculate many of the parameters used in execution. However, Neptune will make assumptions about the probability two homologous bases are different and use fixed thresholds for signature filtering.

```
neptune
  --inclusion inclusion/
  --exclusion exclusion/
  --output output/
```

4.2 Parallel Aggregation

The execution time of Neptune can be greatly improved by enabling parallel k-mer aggregation. The following example will run Neptune with this parallelization enabled.

```
neptune
  --inclusion inclusion/
  --exclusion exclusion/
  --output output/
  --parallelization 3
```

This example will create $4^3 = 64$ parallel k-mer aggregation instances. It is not recommended to create many more instances than there are processing nodes which may be employed by Neptune at one time.

4.3 File Locations

You may wish to specify particular files used in signature discovery. This may be important when specifying references for signature extraction.

```
neptune
  --inclusion inclusion_dir/ in1.fasta in2.fasta
  --exclusion exclusion_dir/ ex1.fasta ex2.fasta
  --reference in1.fasta in2.fasta
  --output output/
```

4.4 DRMAA Parameters

It may be necessary to specify DRMAA parameters to accommodate Neptune job scheduling. This example specifies the resources required by all jobs (**--defaultSpecification**) and further specifies that *k*-mer aggregation jobs (**--aggregateSpecification**) will require more memory. The remaining Neptune parameters are automatically calculated.

```
neptune
  --inclusion inclusion/
  --exclusion exclusion/
  --output output/
  --defaultSpecification "-l h_vmem=6G -pe smp 4"
  --aggregateSpecification "-l h_vmem=10G -pe smp 4"
```

5 Output

Neptune's output directory contains the following items:

- **candidates**: directory containing signature candidates
- **filtered**: directory containing filtered candidates in extracted order
- **sorted**: directory containing filtered signatures in sorted order
- **database**: directory containing Neptune's constructed databases
- **aggregate.kmers**: file containing all observed *k*-mers
- **receipt.txt**: file containing Neptune's run receipt

A file with the same name as each reference will be placed in each output directory, corresponding to the reference file from which it was derived.

5.1 Candidate Signatures

The candidate signatures are the sequences produced from the signature extraction step. These signatures will relatively sensitive, but not necessarily specific. This is because signature extraction is done using exact *k*-mer matches. The candidate signatures are guaranteed to contain no more exact matches with any exclusion *k*-mer than specified by the **--exhits** parameter. However, there may be inexact matches with exclusion targets.

5.2 Filtered Signatures

The filtering step is designed to remove signatures which are not interesting enough to warrant further investigation, because the negative component of their score is prohibitively large. The filtering step removes signatures that align sufficiently with any exclusion target. The filtered signatures are a subset of the candidate signatures.

5.3 Sorted Signatures

The sorted signatures files are organized as FASTA records containing the same signatures as their filtered signatures counterparts. However, the signatures are listed in descending order by their signature score. Signatures are assigned a score corresponding to their highest-scoring BLAST alignments with all inclusion and exclusion targets, which is a sum of a positive inclusion component and a negative exclusion component. This score is maximized when all inclusion targets contain a region exactly matching the entire signature and there exists no exclusion targets that match the signature. The signatures have the following format:

```
>[ID] [SCORE] [IN SCORE] [EX SCORE] [LENGTH] [REF] [POS]
[SEQUENCE]
```

```
>425 score=0.86 in=0.98 ex=-0.13 len=31 ref=ecoli pos=160
TGTCATTCTCCTGTTCTGCCTGTATCACTGC
```

Where:

- [**ID**]: a run-unique ID assigned to the signature
- [**SCORE**]: the total signature score
- [**IN SCORE**]: positive inclusion component of signature score
- [**EX SCORE**]: negative exclusion component of signature score
- [**LENGTH**]: signature length in bases
- [**REF**]: name of the contig from which the signature was extracted
- [**POS**]: starting position of the signature in the reference
- [**SEQUENCE**]: sequence content of the signature

5.4 Databases

The databases directory contains BLAST databases constructed from the inclusion and exclusion files.

5.5 Aggregate k-mers

The aggregated *k*-mers file, aggregated.kmers, contains a list of all *k*-mers observed in the inclusion and exclusion groups. These *k*-mers are sorted and followed by two integers: the number of inclusion and exclusion targets the *k*-mer appears in, respectively.

5.6 Run Receipt

The run receipt contains information about the Neptune execution. It contains a list of all the files in the inclusion and exclusion group, and the command line parameters used for the execution.