NSGW
(Network Sequence Generator Wrapper)
Kappa Release

Monique M. Morin

Department of Computer Science
University of New Mexico

March 8, 2008

Contents

1 Introduction

This is a short write-up on the NSGW code which is currently available for download at http://www.cs.unm.edu/∼morin/ and is released under GNU General Public License (GNU GPL).[1]

The primary function of this code is to act as an interface between NG (our simulator for creating phylogenetic networks) and SEQ-GEN.[2] When all three executables (NG, NSGW, SEQ-GEN), are used together as NETGEN which is described in the application note[3] one can simulate phylogenetic networks (with diploid or polyploid hybrids).

This software is command line driven, written in C, and developed in a Debian Linux environment. Its operation on/in other platforms/environments has not be tried. Theoretically, interested parties should be able to use this software on any Linux/Unix platform which has a standard C/C++ compiler. Please cite the above mentioned application note and/or the download web page when using this software.

Caveat: The NSGW code is currently available as source from http://www.cs.unm.edu/∼morin/. The software is under development as part of a research effort and is offered only "As-Is." There is no guarantee, written or implied, of the software being bug-free or reliable and no liability related to this software will be accepted.

2 Overview

NSGW software was designed and coded to act as the interface for requesting/receiving sequences from the SEG-GEN software during the execution of NETGEN. If one

[1] see http://www.gnu.org/licenses for license details

[2] Rambaut A. and Grassly N. C., **Seq-Gen: an application for the Monte Carlo simulation for DNA sequence evolution along phylogenetic trees**, Comput Appl Biosci 12, 235-238, 1997. See also http://evolve.zoo.ox.ac.uk for further information.

[3] Morin and Moret, "NetGen: generating phylogenetic networks with diploid hybrids", Bioinformatics, Vol 22 Number 15, p1921-1923 August 2006.

desires a different sequence generator, it is this NSGW code which needs to be modified and customized.

NSGW was designed to operate "during" the NETGEN simulation, and the typical user needs to know very little about the NSGW code[4]. During the development and testing process, it was necessary to develop an "after" mode where a complete network topology (without sequences) could be inputted and NSGW would then execute the calls to SEQ-GEN for assigning/evolving the sequences on the predefined topology.

While never intended for general use, it has come to our attention that this "after" capability may have some utility for researchers, hence this separate document is intended to give the essentials for using the NSGW code in this manner. One does not need to concern themselves with this document unless they are interested in using the "after" mode of NSGW.

As described above, the code is designed to run in two primary modes – "during" and "after." While we list all possible execution command lines later in the appendix, our focus here is the "after" mode and all further references to NSGW assume this mode of execution, unless otherwise specified.

NSGW will generate sequences for all nodes present in the inputted topology, including hybrids (diploid and polyploid) and an outgroup if present. Outgroups in NSGW are processed using parameters provided by NETGEN. Hence outgroups are processed by NETGEN and NSGW identically. However, with the case of hybrids, the processing by NETGEN and NSGW is inherently different. NETGEN uses intermediate sequence updates (provided by NSGW – DURING) as part of its hybridization decisions throughout the simulation. NSGW is limited to a predefined topology which was generated without the knowledge of sequences. Hence, networks containing hybrids will have different sequence results depending on whether their sequences are generated using NETGEN/NSGW–DURING or

[4]**NetGen Release Notes, Epsilon Version (Phylogenetic Network Generation Application)**, Technical Report, TR-CS-2006-05, University of New Mexico, Albuquerque, New Mexico; also see http://www.cs.unm.edu/∼morin/ and a copy of the latest release notes for both NETGEN and NSGW are included in the download

NSGW–AFTER.

Both diploid and polyploid hybridizations involve the merging of two lineages into a hybrid node which then continues as its own lineage. In the case of diploid hybridization, the number of homologous chromosomes (and strands in each) belonging to the hybrid stays the same as the parents – hence each parent randomly provides half of its DNA sequences. In the case of polyploid hybridization, the hybrid node receives all of the sequences from both parents. The inputted topology specifies the type of hybrid(s) and the NSGW code deals with the sequence inheritance accordingly.

In general, the NSGW code requires fewer updates than the NG code. However, each new release of either set of source code is considered a new version of NETGEN and re-named appropriately. Also, in order to run this code one needs a copy of the sequence generator, SEQ-GEN, either available from our web page or http://evolove.zoo.ox.ac.uk.

3 Execution

As this code is primarily intended to be used as an interface ("during") and not directly by the user at the command line ("after"), the command line invocations are quite rigid and there is minimal parameter checking/error messages provided by the code. For the sake of completeness, the appendix lists all acceptable forms of the command line invocation, however here we focus on the one needed for the "after" runs.

- NSGW -a -z <int> <inputfile> "<seq-gen-options>"

For example:

- NSGW -a -z 57 nsgw_simple.in "-mHKY"

All output and debugging/error statements are directed to STDOUT and STDERR respectively, therefore one may want to use linux redirection commands for these outputs. For example:

- NSGW -a -z 57 nsgw_simple.in "-mHKY" 1> nsgw_simple.out 2> nsgw_simple.err

The `-a` signals "after" mode and `-z` followed by an integer is for specifying a random number seed for the run. The input file name is the next argument and the format of which is discussed in the next section. The last argument is a listing of options for SEQ-GEN enclosed in quotes. These options customize the SEQ-GEN run (e.g. evolutionary model). At a minimum, one must specify the model to be used as is done in the example above (note that there is no space between the `m` and the model name). For legal SEQ-GEN command line options, one should consult the SEG-GEN material.

Sample input and output files with the above names are included in the download. The file se_out_seq-gen_*pid* is generated and contains output from the most recent call to SEQ-GEN and will contain the version number of SEQ-GEN utilized. This file can be safely deleted after the run is completed.

4 Input/Output

The input for NSGW is very rigid. For ease of development purposes the input for NSGW was assumed to be the output from the NG executable without sequences. It is possible to create an input routine from scratch if desired and the following describes the key segments of such a file and a sample file is listed.

- Comment lines, those starting with `c`, are optional and are ignored by the input routine.
- The first required line starts with an `s` and is followed by summary information about the network (the number of nodes, arcs, leaves, and hybrids, plus the outgroup node id, evolutionary height, min outgroup difference, max outgroup difference and the max number of outgroup tries). This is needed so the overhead processing of the network is completed correctly before the rest of the input is read in.

- The next section lists the internal nodes, denoted by `r` for the root and `n` for the others. Each line contains the node id, its hybrid status (0 = regular node/no hybrid, 1 = diploid hybrid, 2 = polyploid hybrid), number of homologous chromosomes and strands for that node, as well as the sequence length. The final four fields are not used by NSGW but must be present (junk values are fine) as the correct operation of the input routine relies on these values being present.

- Next is a listing of non-extant leaves – these are leaves that died off during the simulation, prior to the end of the simulation (these lines start with an `l`). Subsequently, the extant leaves whose lines start with `e`. All leaves need their id, hybrid status (which should always be 0), number of homologous chromosomes and strands, and sequence length for each string. For these lines, a final value which is typically `time created` must be present, but is not used by NSGW.

- If an outgroup is present, it is listed next with an `o` at the beginning of the line and the remainder is like any other leaf described above.

- Finally branches, `b`, (between two internal nodes) and tips `t`, (from internal node to leaf) are listed last. For these, the parent and child ids are listed as well as branch lengths. The first branch length is the evolutionary branch length which is the value NSGW will pass to SEQ-GEN when requesting a sequence, while the second value is a clock branch length which is reported by NG and must be present as input, but is not used.

```
s 17 17 8 1 16 0.578156 7 10 10
c
r 0 0 1 2 10 0.0000 1.0000 0.2000 0.5000
n 1 0 1 2 10 0.0000 1.0000 0.2000 0.5000
n 2 0 1 2 10 0.1746 1.0000 0.2000 0.5000
n 3 0 1 2 10 0.1490 1.0000 0.2000 0.5000
```

```
n 5 0 1 2 10 0.1746 1.0000 0.2000 0.5000
n 6 0 1 2 10 0.2917 1.0000 0.2000 0.5000
n 7 1 1 2 10 0.1746 1.0000 0.2000 0.5000
n 10 0 1 2 10 0.4305 1.0000 0.2000 0.5000
n 13 0 1 2 10 0.4365 1.0000 0.2000 0.5000
l 8 0 1 2 10 0.2353
c
e 4 0 1 2 10 0.5782
e 9 0 1 2 10 0.5782
e 11 0 1 2 10 0.5782
e 12 0 1 2 10 0.5782
e 14 0 1 2 10 0.5782
e 15 0 1 2 10 0.5782
c
b 0 1 0.000000 0.000000
b 0 16 0.578156 0.578156
b 1 2 0.174590 0.174590
b 1 3 0.149010 0.149010
b 2 6 0.117096 0.117096
b 2 7 0.000000 0.000000
t 3 4 0.429146 0.429146
b 3 5 0.025580 0.025580
b 5 7 0.000000 0.000000
t 5 9 0.403566 0.403566
b 6 10 0.138838 0.138838
t 6 11 0.286470 0.286470
b 7 8 0.060705 0.060705
t 10 12 0.147632 0.147632
b 10 13 0.006013 0.006013
t 13 14 0.141619 0.141619
```

```
t 13 15 0.141619 0.141619
```

The output file produced by NG contains comment lines describing the output. This can be used as a guide for making appropriate input files for NSGW. The output of NSGW includes a listing of node and edge information – including the sequences generated. In addition, the modified Newick format (which is based on the well-known Newick format) is generated. We designed this format to be able to accommodate hybrid nodes and to identify extant taxa. For more details on this format, see the *Bioinformatics* note or the release notes for NETGEN.

5 Acknowledgement and Contact

This work has been supported by the National Science Foundation under grants IIS 01-21377, DEB 01-20709, and EF 03-31654.

The author of this software and document can be contacted by the following means :

morin@cs.unm.edu

OR

Monique Morin
University of New Mexico
Department of Computer Science
Mail stop: MSC01 1130
1 University of New Mexico
Albuquerque, NM 87131-0001
USA

A Appendix A: Execution Command Lines

At this time there are 7 ways to invoke the software and the command line pattern must be followed exactly (ie order and spacing of options must be respected).

The possible command lines preceded here by a letter for reference only are:

- **A:** NSGW -d -l <int> <inputfile> "<seq-gen-options>"
- **B:** NSGW -a <inputfile> "<seq-gen-options>"
- **C:** NSGW -d -l <int> -z <int> <inputfile> "<seq-gen-options>"
- **D:** NSGW -a -z <int> <inputfile> "<seq-gen-options>"
- **E:** NSGW -E -d -l <int> <inputfile> <rnd_num_test_file> "<seq-gen-options>"
- **F:** NSGW -F -a <inputfile> <rnd_num_test_file> "<seq-gen-options>"
- **G:** NSGW -G -t <inputfile> <rnd_num_test_file> "<seq-gen-options>"

The following notes are offered for clarification of the above command lines:

- -a vs -d indicates the mode in which the wrapper will be run (either "a" for after (NSGW-A) or "d" for during (NSGW-D) which is interactively with NETGEN); -t in case **G** is a special tree test case and is essentially an 'after' mode
- <inputfile> means to give the name of an input file but without the < > delimiters;
- -l must always be given for the "d" cases only and it must be followed by a space and then an integer to specify the length of the sequences;
- -z is for when a random number seed is provided – it is to be an unsigned long; therefore a value between 0 and $2^{32} - 1$ (which is 4294967295)
- <int> means to provide an integer;

- <double> means to provide a rate in double format (ie a value between 0.0 and 1.0);

- "<seq-gen-options>" means to provide the desired arguments to be given to Seq-Gen when it is called from the wrapper; most likely this will be a type of model e.g. "-mHKY"; if no special SEQ-GEN options are desired, one still must include a model as the new version of SEQ-GEN requires one to be specified; a simple one is "-mHKY"; note that there is no longer a space after the *m*; these options must be SEQ-GEN recognized options

- command lines **A** and **B** are for NSGW-D and NSGW-A being run without provided seeds

- command lines **C** and **D** are for NSGW-D and NSGW-A being run WITH provided seeds

- command lines **E**, **F**, and **G** are for testing purposes, and their letter needs to be listed in the command line as shown above

- command line **C** is the one used by NETGEN when it is running with "simultaneous sequences"

- command line **F** is intended for use as testing in Rounds 1 and 2 of this document; this is where NSGW-A is validated against NSG/SG using the 10,000 doubles of the rand_10000_ref; note that in this case, the "z" parameter passed to SG is the index for the 10,000 double array, so if running straight NSG/SG, this is 0, if running through NSGW, it must track and pass the appropriate index to SG

- command line **G** is intended only for use as testing Rounds 3 and 4 validation tests of the NETGEN document; this is testing trees generated by NETGEN and then the topology part is used as input for sequence generation into NSGW-A to compare the two sets of sequences; unlike the other rand_num_test_file this one is a file of integers and the number of integers

contained in the file must be on the first line of the file so the code knows how many to read (and this value must be less than or equal to 10,000 since the same structure is used for the other testing above with command line F); see NETGEN document for more details on the test

- DEBUGGING/ERROR statements are all redirected to STDERR; hence, if running NSGW-A, the user may wish to redirect stderr using something such as “2> std_err_output” or “2>> std_error_output”;