

PDA - Phylogenetic Diversity Analyzer

PDA Manual

Version 0.5.1 (Apr 2008)

Copyright 2006-2008 by Bui Quang Minh, Steffen Kläre, and Arndt von Haeseler

Bui Quang Minh

Center for Integrative Bioinformatics Vienna,
Max F. Perutz Laboratories,
Dr. Bohr-Gasse 9/6, A-1030 Vienna, Austria.
email: `minh.bui (at) mfpl.ac.at`

Steffen Kläre

Center for Integrative Bioinformatics Vienna,
Max F. Perutz Laboratories,
Dr. Bohr-Gasse 9/6, A-1030 Vienna, Austria.
email: `steffen.klaere (at) mfpl.ac.at`

Arndt von Haeseler

Center for Integrative Bioinformatics Vienna,
Max F. Perutz Laboratories,
Dr. Bohr-Gasse 9/6, A-1030 Vienna, Austria.
email: `arndt.von.haeseler (at) mfpl.ac.at`

Legal Stuff

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

Contents

1	Introduction	3
1.1	Methods	4
2	Command-line options	4
2.1	General options	5
2.1.1	<file_name> option	5
2.1.2	<output_file> option (NEW!)	5
2.1.3	-k <num_taxa>, -k <min>:<max>, and -k <min>:<max>:<step> option	5
2.1.4	-o <taxon> option	6
2.1.5	-e <file> option	6
2.1.6	-i <file> option	6
2.1.7	-a, -all option	7
2.1.8	-lim <max_limit> option (NEW!)	7
2.1.9	-min option (NEW!)	7
2.1.10	-lout option (NEW!)	7
2.1.11	-oldout option (NEW!)	7
2.1.12	-v option	7
2.2	Exclusive options for trees and networks	7
2.3	Options for budget-constraints	8
2.3.1	-u <file> option	8
2.3.2	-b <budget> option	8
2.3.3	-b <min>:<max>[:<step>] option	8
2.4	Options for area analysis (NEW!)	8
2.4.1	-s <taxa_file> option	8
2.4.2	-excl option	9
2.4.3	-endem option	9
2.4.4	-compl <areas> option	9

2.5	Miscellaneous options	10
3	Outputs	10
4	Example usages	11
4.1	Example usages for trees	11
4.2	Example usages for networks	12
5	Howto employ the method for networks	13
6	Installation	13
7	Version History	14
8	Credits	14

1 Introduction

Phylogenetic Diversity (PD), coined by Faith (1992), is a qualitative measure to assess the biodiversity of species based on a phylogeny. Given n taxa (or species) connected in a phylogenetic tree, the *PD* of any subset of taxa is defined as the sum of the branch-lengths of the minimal subtree connecting these taxa. We have recently proposed an extension of *PD* for split networks (or split systems). Given a split system of n taxa, the *PD* of any subset of taxa is equal to the sum of the weights of all the splits separating at least two taxa in the subset. This definition coincides with the tree-based *PD* when the underlying split system turns out to be a tree.

Phylogenetic Diversity Analyzer (PDA) is a software tool to analyze various aspects of *PD* and *PD*-related measures based on both phylogenetic trees and networks. The major features include:

- Determining the taxon set of a given size k with optimal *PD* (maximal or minimal) for both trees and networks. The algorithms implemented include gPDA and pPDA (Minh *et al.*, 2006) for trees, PD-NET for circular split networks.
- Determining the taxon set with optimal *PD* (maximal or minimal) under budget constraints for both trees and networks. The algorithms implemented include PD-NET-BUDGET.
- Computing the *PD* scores of predefined areas (an area is simply a set of taxa, such as the taxa living in the area). *PDA* also includes the computation of exclusive *PD*, *PD*-Endemism, *PD*-Complementarity.
- Distinguishing between unrooted-*PD* and rooted-*PD*.

PDA was previously named “Phylogenetic Diversity Algorithm”. Since version 0.5 we decided to change it to *Phylogenetic Diversity Analyzer* due to its extended functionalities. An easy-to-use web-interface is now made available at

<http://www.cibiv.at/software/pda/web-pda/>.

Source code is available free of charge under GNU GPL license from

<http://www.cibiv.at/software/pda/>

PDA is written in C++ using Standard Template Library (STL). It will run on most personal computers and workstations if compiled by an appropriate C++ compiler. Please read the *Installation* section 6 for more details.

We suggest that this documentation should be read before using *PDA* the first time. To find out what’s new in the current version please read the *Version History* section 7.

1.1 Methods

The methods are described in detail in the following papers:

- Minh, B. Q., Klaere, S. and von Haeseler, A. (2006) Phylogenetic diversity within seconds. *Syst. Biol.*, **55**, 769–773.
- A manuscript describing the method for split networks is in preparation.
- A manuscript describing the method for budget constraints is in preparation.

2 Command-line options

If you run 'pda -h' or simply 'pda' without any parameters, *PDA* will display a usage screen. The meanings of the options are mainly what you see (For explanations and possible usages see the subsequent sections).

Usage: pda [OPTIONS] <file_name> [<output_file>]

GENERAL OPTIONS:

-h	Print this help dialog. Use -hh to display all options.
<file_name>	User tree in NEWICK format or split network in NEXUS format.
<output_file>	Output file to store results, default is '<file_name>.pda'.
-k <num_taxa>	Find optimal PD set of size <num_taxa>.
-k <min>:<max>	Find optimal PD sets of size from <min> to <max>.
-k <min>:<max>:<step>	Find optimal PD sets of size <min>, <min>+<step>, <min>+2*<step>,...
-o <taxon>	Root name to compute rooted PD, default is unrooted.
-i <file>	File containing taxa to be included into PD set.
-e <file>	File containing branch/split scale and taxa weights.
-a, -all	Identify multiple optimal PD sets.
-lim <max_limit>	The maximum number of PD sets for each k if -a is specified.
-min	Compute minimal PD sets, default is maximal PD sets.
-lout	Also print taxa sets and scores to separate files.
-oldout	Also print output compatible with version 0.3.
-v	Verbose mode.

OPTIONS FOR TREE:

-root	Make the tree ROOTED, default is unrooted.
NOTE: this option and -o <taxon> cannot be both specified.	
-g, --greedy	Run greedy algorithm only.
-p, --pruning	Run pruning algorithm only.
NOTE: by default, the program automatically chooses suitable algorithm.	

OPTIONS FOR SPLIT-NETWORK:

-exhaust Force to use exhaustive search.

NOTE: by default, the program applies dynamic programming algorithm
on circular networks and exhaustive search on general networks.

OPTIONS FOR BUDGET-CONSTRAINT:

-u <file> File containing total budget and taxa preservation costs.

-b <budget> Total budget to conserve taxa.

-b <min>:<max> Find all PD sets with budget from <min> to <max>.

-b <min>:<max>:<step>

Find optimal PD sets with budget <min>, <min>+<step>, <min>+2*<step>,...

OPTIONS FOR AREA ANALYSIS:

-s <taxa_file> Compute PD of areas (user-defined sets) in <taxa_file>.

-excl Compute area exclusive PD.

-endem Compute area endemic PD.

-compl <areas> Compute area PD-complementarity given the listed <areas>.

2.1 General options

2.1.1 <file_name> option

The <file_name> will be the input tree file in NEWICK format or input split network in NEXUS format. The only exception is when you set -r[u] <num_taxa>, the program will generate a random tree and write it into the <file_name> file.

More information on NEWICK tree format can be found at <http://evolution.genetics.washington.edu/phylog/newicktree.html>.

More information on NEXUS file format can be found in the article Maddison *et al.* (1997) or at <http://awcmee.massey.ac.nz/spectronet/nexus.html>.

2.1.2 <output_file> option (NEW!)

This is to set the output file name instead of the default <file_name>.pda where <file_name> is the input file name defined above.

2.1.3 -k <num_taxa>, -k <min>:<max>, and -k <min>:<max>:<step> option

With -k <num_taxa>, *PDA* will compute the optimal *PD* sets of size <num_taxa>. With the new option -k <min>:<max>, *PDA* will compute the optimal *PD_k* sets for *k* from <min> to <max>. So you do not have to run *PDA* several times on the same tree or network for different *k*, and thus save a lot of computational time. It is even more convenient in some

case with `-k <min>:<max>:<step>`: *PDA* will only report the optimal *PD* sets of size k from from `<min>` to `<max>` with the a jumping-step of `<step>`. That means k will iterate through `<min>`, `<min>+<step>`, `<min>+2*<step>`,... until not exceeding `<max>`.

2.1.4 `-o <taxon>` option

From version 0.3, one can distinguish between unrooted and rooted *PD* by this option. See Faith and Baker (2006) for a discussion. If your tree/network has a specific root or outgroup, always specify it by this option. The root will then be always included into the final *PD* set.

2.1.5 `-e <file>` option

The `<file>` containing weights of taxa must be in the following format:

1. First line is a coefficient, which every branch length/split weight should be multiplied with.
2. Each of the rest lines contain taxon name and its weight the “importance” of that taxon.

Any taxa which are not listed in the parameter file will be assigned a weight of ZERO. If you prefer some taxa, you can give them a positive weight. Specify a very high weight to your “favourite” taxa if you want to include them into your final optimal *PD* set.

Please note that the additional coefficient/weights will be incorporated into the resulting *PD* tree / sub-split system, i.e., the final tree / split system will also reflect the coefficient and weights in its branch lengths / split weights.

More information on those additional parameters can be found in Steel (2005).

2.1.6 `-i <file>` option

The file containing all taxa names, which you want to include into your final *PD*-set. The format is simply to list all names separated by blank(s) or new line. NOTE that all names must be corresponding to the user tree file, otherwise an error will be displayed.

This option might be handy in comparative genomics when you have already sequenced several species and have to make a decision what species to be sequenced next. Then the species names, which were already sequenced, can be listed in this file. See Pardi and Goldman (2005) for a discussion.

2.1.7 -a, -all option

This option allows you to identify multiple optimal PD_k sets for a specific k . This is useful in case there are more than one optimal sets with the same PD score. Note that if you specify `-k <min>:<max>`, *PDA* will handle correctly for each k . The `-a` option can be used in conjunction with `-lim <max_limit>` option (see the next sub-section).

2.1.8 -lim <max_limit> option (NEW!)

When you set `-a` option, the number of multiple optimal PD_k sets for each k to be reported will be limited to at most `<max_limit>`. The default limit is 100 if you don't specify. This is simply to avoid *PDA* from memory overflow if millions of such optimal sets exist.

2.1.9 -min option (NEW!)

This option tells *PDA* to find the minimal PD sets instead of the default maximal ones. Note that algorithmically, on trees the greedy algorithm does not work anymore for PD minimization. However, the dynamic programming algorithm presented in Minh *et al.* (2007) can be easily adapted for this case by negating all the branch lengths!

2.1.10 -lout option (NEW!)

This tells *PDA* to write the list of taxa sets into `*.pdtaxa` file and the PD scores into `*.score` file. See Section 3 for more details.

2.1.11 -oldout option (NEW!)

This is for compatibility reason only since by default, version 0.5 only produces the output file `*.pda` which contains more information than only optimal sets, scores, and sub-trees. So this option tells *PDA* to write extra resulting files as outputed in version 0.3. See Section 3 for more details.

2.1.12 -v option

This option tells *PDA* to print more intermediate information while running.

2.2 Exclusive options for trees and networks

The options are easily explainable from the usage screen.

2.3 Options for budget-constraints

From version 0.5 *PDA* is extended to cope with budget constraints. The extended problem is formulated as follows. Given a tree or split network, integer preservation costs c_s for each taxon s and a total integer budget B . Find a subset S of taxa to maximize $PD(S)$ such that the total cost do not exceed the given budget: $\sum_{s \in S} c_s \leq B$. The restriction to integer numbers is not limitation since budgets are normally expressed in integer. If not, it can be easily transformed into integer. This problem is not solvable by a greedy strategy but by a dynamic programming algorithm. A paper describing this is still in preparation.

2.3.1 -u <file> option

This file is in the following format. The first line contains the total integer budget. Each of the subsequent lines contains a taxon name and an associated integer cost, separated by blank(s). Note that any taxon which are not given a cost will be automatically assigned a cost of ZERO.

2.3.2 -b <budget> option

If you don't want to use the budget written in the file specified by -u <file> option, use this option. The budget specified here will be taken for later analysis.

2.3.3 -b <min>:<max>[:<step>] option

This has the same effect as described for -k <min>:<max>[:<step>] option (see Section 2.1.3).

2.4 Options for area analysis (NEW!)

From version 0.5, *PDA* is capable of computing the *PD* scores of several areas. An area simply refers to a user-defined subset of taxa. It also can compute the exclusive *PD*, endemic *PD*, and *PD*-complementarity of areas. See the subsequent sections for more details. Note that this has nothing to do with *PD* optimization. It simply helps you analyze the scores of different areas you are currently interested in.

2.4.1 -s <taxa_file> option

The list of areas is given in <taxa_file>. This file can be in one of the two formats: simple text file or NEXUS format. *PDA* will automatically detect the type of this file.

- Simple text file: The first line is the number of taxa n_1 of the first area. The next n_1 lines are the names of n_1 taxa in the first area. Then comes the number n_2 (the number of taxa of the second area) and n_2 lines storing the names of those taxa in the second area. This repeats until the last area or reaching the end of file.
- NEXUS format: *PDA* will read the SETS block of the NEXUS file. For simplicity, here is an example:

```
#nexus

begin sets;
  taxset 'a1' = 'a' 'b' 'c';
  taxset a2 = c d g n;
  taxset a3 = h i;
  taxset a4 = j k;
  taxset a5 = l m;
  taxset a6 = h i;
  taxset a7 = j k;
  taxset a8 = l m;
end; [sets]
```

2.4.2 -excl option

This tells *PDA* to compute the exclusive *PD* of all areas as well. In short, given X the set of all taxa and A an area, the exclusive *PD* of A is simply: $ePD(A) = PD(X) - PD(X - A)$. See Lewis and Lewis (2005) for the original description of this measure.

2.4.3 -endem option

This tells *PDA* to compute the endemic *PD* of all areas as well. Given X the set of all taxa and $A_1, A_2, \dots, A_m$ all areas you have. Let U be the union taxon set of all areas. Then the endemic *PD* of a particular area A_i is: $PD(A_1 \cup \dots \cup A_m) - PD(A_1 \cup \dots \cup A_{i-1} \cup A_{i+1} \cup \dots \cup A_m)$. See Faith *et al.* (2004) for more details and interpretation.

2.4.4 -compl <areas> option

This tells *PDA* to compute the *PD*-complementarity of all areas given the list <areas>. The list can contain one area name or several area names separated by commas. Let B be the union taxon set of all given areas. Then the *PD*-complementarity of a particular area A_i is: $PD(A_i|B) = PD(A_i \cup B) - PD(B)$. See Faith *et al.* (2004) for more details and interpretation.

2.5 Miscellaneous options

If you run 'pda -hh', *PDA* will display some more features:

GENERATING RANDOM TREES:

```
-r <num_taxa>      Create a random tree under Yule-Harding model.  
-ru <num_taxa>     Create a random tree under Uniform model.  
-rcat <num_taxa>   Create a random caterpillar tree.  
-rbal <num_taxa>   Create a random balanced tree.  
-rcsg <num_taxa>   Create a random circular split network.  
-rlen <min_len> <mean_len> <max_len>  
                  minimum, mean, and maximum branch lengths of the random trees.
```

MISCELLANEOUS:

```
-d <sample_size>   Compute PD distribution of random sets of size k.  
-dist <outfile>    Calculate the distance matrix inferred from tree.  
-seed <number>     Set the seed for random number generator.  
-con <treefile>    Compute an extended majority-rule consensus tree.  
-conet <treefile>  Compute a consensus network.  
-sup <treefile>    Assign support values of each node in input tree.
```

The usage is explained there. Feel free to use!

3 Outputs

From version 0.5 all outputs will be written to `<file_name>.pda` by default or `<output_file>` if you specify it in the command-line.

If you specify `-1out`, all the taxa sets are additionally written to `<file_name>.pdtaxa` and the *PD* scores are written to `<file_name>.score`.

If you specify `-oldout`, additional files are written as of version 0.3 as follows.

Resulting *PD* taxa set will be written into: `<file_name>.<k>.pdtaxa`

If the option `-a` or `-all` is specified and multiple optimal is observed, subsequent optimal taxa sets will be written into: `<file_name>.<k>.pdtaxa.1`,
`<file_name>.<k>.pdtaxa.2,...`

The score is printed to `<file_name>.score`.

For tree, resulting sub-trees are written into:

- If you specify `-b` or `--both`:

<file_name>.<k>.greedy for greedy algorithm, and
<file_name>.<k>.pruning for pruning algorithm.

- Otherwise: <user_tree>.<k>.pdtree.

NOTE:

- Two options `-lout` and `-oldout` are mutually exclusive. That means if you specify both, the later specified option in the command-line will override the earlier one.
- For the case of split network, no resulting sub-network is written.
- If you choose option to generate a random tree/network, it will be written to <file_name> (it acts as output instead of the input file).

4 Example usages

4.1 Example usages for trees

```
./pda test.tree -k 4
```

Infer the maximal *PD*-tree of 4 taxa from the tree in `test.tree` (in NEWICK format). *gPDA* or *pPDA* algorithm (Minh *et al.*, 2006) will be determined automatically. Resulting tree will be written to `test.tree.4.pdtree`. Resulting taxa set will be printed to `test.tree.4.pdtaxa`.

NOTE: The program will automatically detect the type of the input file (either NEWICK or NEXUS) to apply appropriate *PDA* algorithms. It should not depend on the file name (`.tree` or `.nex` does not matter).

```
./pda test.tree -k 4 -o c
```

Compute the “rooted *PD*”, the tree is rooted at taxon ‘c’. ‘c’ will be included into the final *PD*-set. See section 2.1.4 for more detail.

```
./pda test.tree -k 4 -g
```

Same as the first command, but only apply the *gPDA* algorithm.

```
./pda test.tree -k 4 -b
```

Run both algorithms. Resulting trees will be written into `test.tree.4.greedy` and `test.tree.4.pruning`.

```
./pda test.tree -k 4 -e test.pam
```

Read the weight information from `test.pam` file (more detail in section 2.1.5) and integrate this into the tree in `test.tree`. Then run the program as the first example command.

```
./pda test.tree -k 4 -i test.taxa
```

Include the “favourite” taxa listed in `test.taxa` (more detail in section 2.1.6) into the final *PD*-set.

```
./pda test.tree -k 4 -e test.pam -i test.taxa
```

Combining both features of the above two example commands.

```
./pda 1000.tree -r 1000
```

Generate a 1000-taxa random tree under Yule Harding Model. Write resulting tree into `1000.tree` file under NEWICK format.

4.2 Example usages for networks

```
./pda test.nex -k 3
```

Find the maximal PD_3 set of the split network in `test.nex` (in NEXUS format, as produced by e.g., SplitsTree 4 program (Huson and Bryant, 2006)). *PDA* will detect whether the input split system is circular or not. If yes, apply the dynamic programming algorithm, otherwise, use exhaustive search. Resulting taxa set will be printed to `test.nex.3.pdtaxa`.

```
./pda test.nex -k 3 -o 2
```

Compute the “rooted *PD*”, the split system is rooted at taxon ‘2’. ‘2’ will be included into the final *PD*-set. See section 2.1.4 for more detail.

NOTE: With this option, the program will normal perform much faster (the time-complexity reduces by a factor of n , where n is the number of taxa). So always specify `-o <taxon_name>` if you are sure that some taxon must be present in the final *PD* set (e.g. the taxon with a very long terminal split).

Other basic options (`-i`, `-e <file>`) should also work fine with split network.

```
./pda test.nex -k 4 -mk 2
```

Identify all optimal *PD* sets containing 2 to 4 taxa. The resulting *PD* sets will be printed to `test.nex.2.pdtaxa`, `test.nex.3.pdtaxa`, `test.nex.4.pdtaxa`. The *PD* scores are written to `test.nex.score` containing several lines. Each line as `<sub_size> <corresponding_score>`, where `<sub_size>` should go from 2 to 4.

```
./pda test.nex -k 3 -all
```

Find all multiple optimal PD_3 sets: if there are more than 1 optimal 3-set, all of them will be printed. The second optimal set will be in `test.nex.3.pdtaxa.1`, the third in `test.nex.3.pdtaxa.2`, etc.

NOTE: This optimal might lead to exponential computing time, as it actually depends on the number of optimal *PD* sets!

```
./pda test.nex -k 4 -mk 2 -all
```

Combine the features of the two previous commands.

5 Howto employ the method for networks

A way to employ this new feature is to use together with program SplitsTree 4 (Huson and Bryant, 2006), available on the website <http://www.splitstree.org/>. First, you reconstruct a circular network by e.g., Neighbor-net method (Bryant and Moulton, 2004). The resulting network is then saved to a NEXUS file, e.g., `mynet.nex`. Then you can feed `mynet.nex` directly to *PDA*.

There could be several blocks in `mynet.nex` input file. However, *PDA* only cares for `TAXA` and `SPLITS` blocks. All other will be ignored, including `CHARACTERS`, `DISTANCES`, `NETWORKS`, `ST_ASSUMPTIONS`, etc. You can also prepare your own split network. A simple input file is inside the `src/` folder under the name `test.nex`.

NOTE: The algorithm for circular network is very fast. So always specify the `CYCLE` command inside the `SPLITS` block of the NEXUS file. Otherwise, an exhaustive search will be applied and very slow.

6 Installation

To build *PDA* from the sources you need a functional C++ compiler installed (This is usually the case on UNIX/Linux systems. For Windows you might want to obtain CygWin or XCode for MacOSX). Then you can follow the procedure below:

1. Download the current version of the software (`pda-XXX.tar.gz` where XXX is the current version number) from its web page <http://www.cibiv.at/software/pda/>.
2. Extract the files (e.g., with `tar xvzf 'pda-XXX.tar.gz'` under Unix) This should create a directory `pda-XXX`.
3. Change into this directory.
4. To compile the program, type the following:

```
./configure
```

This should configure the package for the build. You might also want to refer to the `INSTALL` file for more (general) details.

```
make
```

This compiles and builds the executable '`pda`' (or '`pda.exe`' on Windows systems) to be found in the '`src`' directory. This executable can be copied to your system's search path such that it is found by your system or it can be installed to the default destination (e.g., `/usr/local/bin` on UNIX/Linux) using

```
make install
```

If you encounter problems, please ask your local administrator for help.

7 Version History

0.5.1 Some bugs fixed and codes cleaned. A new user manual.

0.5 Extension to budgeted *PD*. Being able to compute *PD* and *PD*-related measures for areas.

0.3 Extension to split networks. Distinguish between unrooted and rooted *PD*. Print also the taxa set now.

0.21 Fix a minor bug with STL vector constructor while compiling.

0.2 Inclusion of `-i <file>` option.

0.1 Initial version.

8 Credits

The parser for NEXUS file format is derived from the Nexus Class Library (Lewis, 2003).

Acknowledgement

The authors would like to thank Dan Faith for helpful suggestions on the subject and Heiko Schmidt and Tanja Gesell for fruitful discussions. We also thank Mike Steel for constructive comments. Financial support from the Wiener Wissenschafts-, Forschungs- and Technologiefonds (WWTF) is greatly appreciated.

References

- Bryant, D. and Moulton, V. (2004) Neighbor-net: An agglomerative method for the construction of phylogenetic networks. *Mol. Biol. Evol.*, **21**, 255–265.
- Faith, D., Reid, C. and Hunter, J. (2004) Integrating phylogenetic diversity, complementarity, and endemism for conservation assessment. *Conserv. Biol.*, **18**, 255–261.
- Faith, D. P. (1992) Conservation Evaluation and Phylogenetic Diversity. *Biol. Conserv.*, **61**, 1–10.
- Faith, D. P. and Baker, A. M. (2006) Phylogenetic diversity (PD) and biodiversity conservation: Some bioinformatics challenges. *Evolutionary Bioinformatics Online*, **2**, 70–77.
- Huson, D. H. and Bryant, D. (2006) Application of phylogenetic networks in evolutionary studies. *Mol. Biol. Evol.*, **23**, 254–267.
- Lewis, L. A. and Lewis, P. O. (2005) Unearthing the molecular diversity of desert soil green algae. *Syst. Biol.*, **54**, 936–947.
- Lewis, P. O. (2003) NCL: a C++ class library for interpreting data files in NEXUS format. *Bioinformatics*, **19**, 2330–2331.
- Maddison, D. R., Swofford, D. L. and Maddison, W. P. (1997) NEXUS: An extensible file format for systematic information. *Syst. Biol.*, **46**, 590–621.
- Minh, B. Q., Klaere, S. and von Haeseler, A. (2006) Phylogenetic diversity within seconds. *Syst. Biol.*, **55**, 769–773.
- Minh, B. Q., Klaere, S. and von Haeseler, A. (2007) Phylogenetic diversity on split networks. *Technical Report NI07090-PLG, Isaac Newton Institute, Cambridge, UK*.
- Pardi, F. and Goldman, N. (2005) Species choice for comparative genomics: Being greedy works. *PLoS Genet.*, **1**, 672–675.
- Steel, M. (2005) Phylogenetic diversity and the greedy algorithm. *Syst. Biol.*, **54**, 527–529.