

PDA - Phylogenetic Diversity Algorithm

PDA Manual
Version 0.3 (Apr 2007)

Copyright 2006-2007 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	1
2	Methods	2
3	Command-line options	2
4	Example usages	3
4.1	Example usages for trees	3
4.2	Example usages for networks (NEW!)	4
5	Notes on several options	4
5.1	<file_name> option	4
5.2	-o <taxon> option (NEW!)	5
5.3	-e <file> option	5
5.4	-i <file> option	5
6	Outputs	5
7	Howto employ the method for networks (NEW!)	6
8	Tips and tricks (NEW!)	6
8.1	How to compute the PD score of a user-defined taxa set?	6
8.2	How to identify multiple optimal PD sets on trees?	6
9	Installation	7
10	Version History	7
11	Credits	8

1 Introduction

Given n taxa (e.g. species) connected in a phylogenetic tree or a split network, this software computes a subset of k taxa with maximal phylogenetic diversity (PD). For trees the PD of any subset of taxa is defined as the sum of the branch-lengths of the minimal subtree connecting the taxa in the subset (Faith, 1992). For split networks we extend the PD notion as the sum of the weights of all the splits which separate at least two taxa in the subset.

For trees the program implemented two efficient algorithms (gPDA, pPDA; Minh *et al.*, 2006) based on the greedy strategy which always finds the maximal PD set of a given size k (Steel, 2005; Pardi and Goldman, 2005). For split networks we have recently proposed a dynamic programming algorithm for so-called circular split networks, a commonly used class of networks. Such a network can be reconstructed by e.g. the Neighbor-net method (Bryant and

Moulton, 2004). For general networks the software uses an exhaustive search which is only applicable if the subset size k is small.

PDA is available free of charge 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 9 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 10.

2 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.

3 Command-line options

Usage: ./pda [OPTIONS] <file_name>

BASIC OPTIONS:

-h	Print this help dialog.
<file_name>	User tree in NEWICK format or split network in NEXUS format.
-k <num_taxa>	Number of taxa to be preserved.
-o <taxon>	Root name to compute optimal rooted PD. If not specified, consider tree/network as unrooted.
-i <file>	File containing taxa to be included into PD set.
-e <file>	File containing branch/split coefficient and taxa weights.

OPTIONS FOR TREE:

-g, --greedy	Run greedy algorithm only.
-p, --pruning	Run pruning algorithm only.
-b, --both	Run both algorithms.

NOTE: by default, the program automatically chooses suitable algorithm.

OPTIONS FOR SPLIT NETWORK:

-mk <min_taxa>	Find all PD sets with size from <min_taxa> to <num_taxa>.
-all	Identify multiple optimal PD sets.
-exhaust	Force to use exhaustive search.

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

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 5.2 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 5.3) 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 5.4) 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 (NEW!)

```
./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.pdntaxa`.

```
./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 5.2 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.pdntaxa`, `test.nex.3.pdntaxa`, `test.nex.4.pdntaxa`. 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.pdntaxa.1`, the third in `test.nex.3.pdntaxa.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 Notes on several options

5.1 <file_name> option

The `<file_name>` will be the input tree/network file if you specify to run the algorithm by `<num_taxa>`. Otherwise, if 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/phylip/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>.

5.2 -o <taxon> option (NEW!)

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.

5.3 -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).

5.4 -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.

6 Outputs

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

For split network, if the option `-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: 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 the `<file_name>`.

7 Howto employ the method for networks (NEW!)

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.

8 Tips and tricks (NEW!)

8.1 How to compute the PD score of a user-defined taxa set?

Simply use `-i <file>` option together with `-k <num_taxa>` where `<num_taxa>` is equal to the number of taxa listed in the `<file>`.

8.2 How to identify multiple optimal PD sets on trees?

You might recognize that this feature is not available for trees, but only for networks. So here is a simple way. Open the tree file, e.g., `test.tree` with SplitsTree 4, then save it to another file, e.g., `test.tree.nex`. Then run *PDA* on `test.tree.nex` and *PDA* will consider it as a circular network! Now you can use `-all` option in the command line.

9 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.

10 Version History

- 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.

11 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. 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, 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.
- 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.