TargetP

TargetP

The content of this page is based on information retrieved from TargetP website on December 28, 2010. For more details and updates consult the original documentation. All code snippets on this page are shell commands, and have been successfully tested on the `ixchel.iplantcollaborative.org` *Linux server.*

TargetP is a tool for predicting subcellular location of eukaryotic proteins. The location assignment is based on the predicted presence of any of the N-terminal presequences: chloroplast transit peptide (cTP), mitochondrial targeting peptide (mTP) or secretory pathway signal peptide (SP).

Availability

Online use

TargetP is publicly available for server-side execution through its website.

Offline use

TargetP is available as a standalone tool for *academic users only*. Other users are requested to contact CBS Software Package Manager at software@cbs.dtu.dk.

Download

To obtain the tool, one must fill an online form. The information required includes:

- the target platform (Unix-based, including Linux);
- the user's name, position, email, and affiliation;
- acceptance of the terms and conditions.

Upon submission of the form, the tool is *sent* to the provided email address. (There seems to be no way to actually *download* the tool from the website, so no link is provided here.) The tool is sent as a zipped tar archive of size under 1 MB.

The distribution archive obtained on December 28, 2010 can be found in the attachments to this page.

Installation

The installation procedure consists of:

- unpacking the archive, using the command line tools `gzip` and `tar`;
- editing a bunch of settings *within* the executable shell script;
- *manually* moving files into their destination directories.

There is no automated configuration and installation procedure (neither `configure` nor `makefile` are included); there is in fact no compilation involved, the tool is provided as a few precompiled binaries and a collection of Perl and Awk scripts, all called via a customizable shell script.

The following is a brief summary of the installation procedure used to setup `targetp` on `xchel`. Consult the readme file included in the package (also available online) for more details.

```
# full path to the downloaded file; adjust
TARGETP="~/install/targetp/targetp-1.1b.Linux.tar.Z"

# path to where targetp is to be installed; adjust
BASE=~/opt/targetp-1.1

# unpack the archive
if [ -e "$BASE" ]; then rm -Rf "$BASE"; fi
mkdir -p "$BASE"
cd "$BASE"/..
tar xzf "$TARGETP"
cd targetp-1.1

# adjust settings within the shell script; use your favorite settings
TEMP="temp.$$"
sed -r "
   s|^(TARGETP=).*|\1$BASE|
   s|^(TMP=).*|\1/tmp|
   s|^(CHLOROP=).*|\1$BASE/chlorop|
   s|^(SIGNALP=).*|\1$BASE/signalp|" targetp > "$TEMP"
mv "$TEMP" targetp
chmod 755 targetp

# test the installation
(./targetp -P test/one.fsa && ./targetp -P test/twelve.fsa) > output.txt
if [ $? -ne 0 ]; then echo "targetp: testing failed"; fi

# install the manpage; adjust the path; you may need administrator privileges
MANPATH="/usr/local/man/man.1"
sudo cp targetp.1 "$MANPATH"
man targetp > /dev/null
if [ $? -ne 0 ]; then echo "targetp: manpage installation failed"; fi

# add the targetp directory to search path; adjust
echo "export PATH=\"\$PATH:$BASE\"" >> ~/.bashrc
```

Notes:

- `targetp-1.1b.Linux.tar.Z` should be replaced with the actual name of the received file.
- You may want to use `$(tempfile)` to generate a temporary file name; `xchel` does not provide `tempfile`, hence `temp.$$` is used here.

Dependencies

TargetP has two types of dependencies: those assumed to be pre-installed on the installation platform, and those obtainable from the same source as TargetP.

According to the distributed readme file, the former dependencies include `uname`, `paste`, `echo`, `perl`, `awk`, and `sh`. The first three are regular components of GNU coreutils. The other three are are also typical components found on Unix platforms. No requirement is specified as to the version of any of the above.

The latter, optional dependencies are the programs `chlorop` and `signalp`, obtainable from the developers the same way as TargetP. Both should be installed in the directory referred to by $BASE above. (Alternatively, the procedure should be adapted accordingly to a different choice. See the corresponding readme files for details.) The distribution archives for both `chlorop` and `signalp`, obtained on December 28, 2010 can be found in the attachments to this page.

Version

The explicit version number of the archive obtained on December 28, 2010 and used for the purpose of integration into iPlant CI is 1.1; `targetp`, the main shell script lists 2001 as the most recent development date for that version number. However, other files in the archive bear dates spanning the years 2001-2007, and it's rather difficult to judge whether any significant modifications to the program as a whole were made since the release of version 1.1 in 2001. When called with the 'print version' parameter `-v`, `targetp` responds with `targetp v1.1b, March 2006`. It appears safe to assume that the developers did not put much attention to coherent versioning.

Usage

The following usage description is based on the available documentation and some preliminary testing. Please consult the documentation for further details.

Command line invocation

The general form of TargetP command line invocation is:

```
targetp [parameters] [input]
```

where `parameters` is a sequence of obligatory and optional parameters, and `input` is a sequence of input file names.

Parameters

For the current purposes, a selection of parameters have been included in the table below. For more details and further parameters, consult the TargetP manual page.

actual command-line parameter	name and brief description of the parameter	required	default value	text, number, or name of file	description of validation rules
-P	use the plant version	Y	none	none; -P is a flag parameter	According to the documentation, either -P (for plants) or -N (for non-plants) must be specified. In fact, if none is specified, TargetP acts as if -N were given.
-p	Pcut	N	none	real number in the range (0.0, 1.0]	"In order to increase the specificity of cTP prediction, use Pcut as a cutoff for predicting cTP: if the winning score is the chloroplast (cTP) score, specifying Pcut means that the score also has to be above that value; if not, the sequence will be left unpredicted, and an asterisk ⭐ will be output in the Loc column." Values lower than or equal to 0.0 give output as if no -p was specified; values greater than 1.0 give output as for 1.0.
-t	Tcut	N	none	real number in the range (0.0, 1.0]	As above, but for mTP prediction.
-s	Scut	N	none	real number in the range (0.0, 1.0]	As above, but for SP prediction.
-h	help	N	none	none; -h is a flag parameter	Display command line syntax and exit.
-v	version	N	none	none; -v is a flag parameter	Display version info and exit.

Input

TargetP accepts one or more input files which have to contain protein sequences in the standard FASTA format. Documentation for the online service states:

> ... there may be not more than 2,000 sequences and 200,000 amino acids in one submission. The maximal allowed sequence length is 4000 amino acids.

Documentation for the offline tool does not mention such constraints. However, there seem to be limitations here too, as the tool produces apparently wrong output (with no complaint) if sufficiently large input is used. (It remains to establish whether the limitations are as for the online tool or are different.)

Example

For the current purposes, TargetP was tested with two FASTA files obtained from UniProt on December 29, 2010. The content of one of them represents 41 reviewed mitochondrial transmembrane transporter proteins in A. thaliana. The content of the other represents 58 reviewed membrane signal transmission proteins, also from A. thaliana. For the full content, see the attached files (`mtt.fasta` and `mst.fasta`, respectively) or run a `curl` or `wget` job (the query URLs are in the attached files `mtt.url` and `mst.url`, respectively).

```
# download sample input files
for NAME in {mtt,mst}; do
   eval "curl -o $NAME.fasta '$(cat $NAME.url)'"
   targetp -P $NAME.fasta > $NAME.out; done
```

It should be noted that the predictions from TargetP are neither complete nor entirely in agreement with the actual annotation of the proteins. (Please read the caveats section in the manual for comments on possible sources of inaccuracy; the original publications are further sources of insight.)

```
# examine sample predictions
for NAME in {mtt,mst}.out; do
   grep '^sp' $NAME | cut -c 58 > predictions
   N=$(wc -l predictions | cut -d ' ' -f 1)
   echo "[$NAME] M: $(grep -c M predictions)/$N, C: $(grep -c C predictions)/$N, S:
$(grep -c S predictions)/$N, unknown: $(grep -cr '[_*]' predictions)/$N"; done
rm predictions
# [mtt.out] M: 12/41, C: 2/41, S: 8/41, unknown: 19/41
# [mst.out] M: 2/58, C: 6/58, S: 10/58, unknown: 40/58
```

Reference

Locating proteins in the cell using TargetP, SignalP, and related tools
Olof Emanuelsson, Søren Brunak, Gunnar von Heijne, Henrik Nielsen
Nature Protocols 2, 953-971 (2007).
PMID:17446895 | DOI:10.1038/nprot.2007.131

Predicting subcellular localization of proteins based on their N-terminal amino acid sequence.
Olof Emanuelsson, Henrik Nielsen, Søren Brunak and Gunnar von Heijne.
J. Mol. Biol., 300: 1005-1016, 2000.
PMID:10891285 | DOI:10.1006/jmbi.2000.3903

ChloroP, a neural network-based method for predicting chloroplast transit peptides and their cleavage sites.
Emanuelsson O, Nielsen H, von Heijne G
Protein Science., 8, 978-984, 1999
PMID:10338008

Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites.
Henrik Nielsen, Jacob Engelbrecht, Søren Brunak and Gunnar von Heijne.
Protein Engineering, 10:1-6, 1997.
PMID:9051728