ATV - A Tree Viewer

Version 3.1 January, 2006

http://www.phylogenomics.us/atv

Zmasek C. M. and Eddy S. R. (2001) ATV: display and manipulation of annotated phylogenetic trees. Bioinformatics, 17, 383-384.

Christian M. Zmasek
cmzmasek@yahoo.com

Ethalinda Cannon
ethy@a415software.com

Table of Contents

1. Overview

ATV (A Tree Viewer) allows the display and manipulation of phylogenetic trees in a variety of formats. These trees may be quite large. Both internal and external nodes (or tips) can have data fields associated with them.

ATV is part of the FORESTER framework, a framework for phylogenomics. ATV can be used both as an application and embedded in a web page as an applet. FORESTER is described in a separate document.

The FORESTER framework is part of a larger system, RIO, Resampled Inference of Othologs, which is based on FORESTER, Pfam, and Hmmer. (http://www.rio.wustl.edu/)

Files for all three (ATV, FORESTER, RIO) are included in the distribution, though this documentation concerns only ATV.

This document will describe how to install and run ATV. Please see the README and INSTALL files in the distribution for the most current information.

Contact: cmzmasek@yahoo.com
ethy@a415software.com

Requirements:	Java:	1.4.x or greater
	OS:	Unix/Linux, any MS Windows which supports Java 1.4 or better

2. Obtaining and Installing ATV

2.1. Obtaining ATV

Since ATV is part of the FORESTER framework it must be used in conjunction with FORESTER. FORESTER (including ATV) can be downloaded at:
http://www.phylogenomics.us/atv/

It is also available on SourceForge:
http://sourceforge.net/projects/forester-atv/

The distribution contains:

- this documentation in `docs/` subdirectory
- all the source code (.java) files (in a directory named `java/src/`)
- ATVapp.jar – the compiled and archived files for the ATV application (requires Java 1.4 or higher)
- ATVapplet.jar – the compiled and archived files for the ATV JApplet (requires Java 1.4 or higher)
- ATVConfig.conf – the UI config file
- build.xml – Ant build file.
- main.txt – manifest file for building jar files.
- ATV.html – a sample html page for displaying ATV as an applet.
- 00README.txt – the most recent information about FORESTER-ATV
- LICENSE.txt – the license for FORESTER-ATV (same as in this documentation)
- INSTALL.txt – current instructions for compiling and deploying FORESTER-ATV
- RELEASE_NOTES – differences between versions
- Sample trees – in directory `examples/`
- ATVapp.bat – an example .bat file (for Windows users)
- There may be additional files which are related to FORESTER and RIO and not needed for ATV. See http://www.rio.wustl.edu/ for more information on RIO.

2.2. Unpacking ATV

Unix/Linux:

```
gunzip forester.tar.gz
```

```
     tar -xvf forester.tar
```

Microsoft Windows:

Use a program such as WinZip to unzip "forester-*version*.zip".

In both cases, a new directory named "`forester-version`" (e.g. `forester-3.1/`) will be created in your working directory. This directory contains all the unpacked files.

2.3. Running ATV

The ATV application and applet can be run directly from the jar files, (ATVapp.jar and ATVapplet.jar), or you can compile the source code.

2.3.1. Compiling the Source Code

Compile the application with (using the proper path deliminators for your operating system):

```
cd ATV-install-directory/java/
javac org/forester/atv/ATVapp.java
javac org/forester/extensions/*.java
```

You can also compile with Ant, using the build.xml file included in the distribution.

To create a jar file with the newly compiled code: (this will jar both the .java and .class files together)

```
jar –cvf ATVapp.jar org/
```

If you want to make it possible to run the jar file by double-clicking its icon, then use the manifest file, main.txt, provided in the distribution.

```
jar –cvmf main.txt ATVapp.jar org/
```

For more information about jar files see:
http://java.sun.com/j2se/1.5.0/docs/guide/jar/index.html

2.3.2. Launching the ATV Application

To run ATVapp from the code:

```
java org.forester.atv.ATVapp [opts] [name of tree file in NH,
NHX, or PhyloXML format]
```

To run ATV from the jar file, either double-click ATVapp.jar or type:

```
cd ATV-install-directory
java -jar ATVapp.jar [opts] [name of tree file in NH, NHX or
PhyloXML format]
```

Command line options:
-c filename – config file; use the indicated config file to configure the app controls. See section 3.3. The Control Panel and Configuation File for more information about config files.

2.3.3. Launching the ATV Applet from a Web Page

The file "ATVapplet.jar" should be placed in the same directory as your HTML and tree files (at least, this is the easiest way, if you are more experienced with applets and HTML, you will know what to do anyway).

Compile the source code files with:

```
javac ATVapplet.java
javac org/forester/extensions/*.java
```

and then use:

```
jar cf ATVapplet.jar org/
```

to create file "ATVapplet.jar" from the resulting .class files.

To use an ATV Applet in your web page, you need to place the following into your HTML file:

```
<APPLET ARCHIVE = "ATVapplet.jar"
 CODE = "org.forester.atv.ATVapplet.class"
 WIDTH = 200 HEIGHT = 50>
 <PARAM NAME = url_of_tree_to_load VALUE = URL of your tree file>
</APPLET>
```

Or, if you have more than one tree, it is easier to use JavaScript. For an example, see http://www.phylogenomics.us/atv/examples/index.html

Define function "openWin(tree file URL)":

```
<SCRIPT language="JavaScript">
<!-- hide
function openWin( u ) {
  atv_window = open("",
                 "atv_window",
                 "width=300,height=150,status=no,toolbar=no,menubar=no,resizable=yes"
                 );

  // open document for further output
  atv_window.document.open();

  // create document
  atv_window.document.write( "<HTML><HEAD><TITLE>ATV" );
  atv_window.document.write( "</TITLE></HEAD><BODY>" );
  atv_window.document.write( "<BODY TEXT =\"#FFFFFF\" BGCOLOR =\"#000000\">");
  atv_window.document.write( "<FONT FACE = \"HELVETICA, ARIAL\">" );
  atv_window.document.write( "<CENTER><B>" );
  atv_window.document.write( "Please do not close this window<BR>as long as
   you want to use ATV." );
  atv_window.document.write( "<APPLET ARCHIVE = \"ATVapplet.jar\"" );
  atv_window.document.write( " CODE = \"org.forester.atv.ATVapplet.class\"" );
  atv_window.document.write( " WIDTH = 200 HEIGHT = 50>\n" );
  atv_window.document.write( "<PARAM NAME = url_of_tree_to_load\n" );
  atv_window.document.write( " VALUE = " );
  atv_window.document.write( " http://" + u + ">" );
  atv_window.document.write( "</APPLET>" );
  atv_window.document.write( "</BODY></HTML>" );

  // close the document - (not the window!)
  atv_window.document.close();
}

// -->
</SCRIPT>
```

Then place buttons which will open ATV applets with the following (please note that URLs to tree files must begin with "http://"):

```
<form>
 <input type=button value="View … tree" onClick="openWin( 'http://………' )">
</form>
```

Applet parameters:

url_of_tree_to_load – set the initial tree to load into ATV.

config_file – set a config file (other than the default, ATVconfig.conf in codebase). See section 4.2. The Control Panel and Configuration File for more information about config files.

2.3.4. Making Life Easier

Add the path to the ATV Java code to your CLASSPATH so that you can type (e.g.) “java ATVapp” from anywhere. For Unix/Linux use:

```
setenv CLASSPATH ./:/path/to/otherjavaclasses:/path/to/ATV
```

For Windows go to: “Start” | “Settings” | “Control Panel” | “System” | “Advanced” | “Environment Variables” and add the path to the ATV code to CLASSPATH.

You can save more typing by this in Unix/Linux: (example)

```
alias atv ‘java ATVapp’
```

Windows users can modify the supplied example .bat file (“ATVapp.bat”) and use it to start ATV with a mouse click on its icon.

Obviously, all these commands are just examples, you need to modify them according to your directory structure and according to which version of ATV you are using and according to whether you are using the Java runtime environment.

3. Formats for Reading and Writing Phylogenetic Trees

ATV can read and write phylogenetic trees a number of different formats: the **New Hampshire** (NH) format as defined by Dr. Joseph Felsenstein, the extended **New Hampshire X** (NHX) format as defined below, and **PhyloXML**, an XML language for describing richly-annotated trees.

Note: Only the application is allowed to read and write from files, the applet is allowed to read only from URLs and cannot write out modified trees.

3.1. The New Hampshire Format (NH)

In following is **Dr. Joseph Felsenstein's definition of the New Hampshire Standard**:

“*(c) Copyright 1990 by Joseph Felsenstein. Written by Joseph Felsenstein. Permission is granted to copy this document provided that no fee is charged for it and that this copyright notice is not removed.*

The New Hampshire Standard for representing trees in computer-readable form makes use of the correspondence between trees and nested parentheses, noticed in 1857 by the famous English mathematician Arthur Cayley. If we have this rooted tree:

```
A                   D
 \           E     /
  \    C    /     /
   \   !   /     /
    \  !  /     /
B    \ ! /     /
 \     o      /
  \    !     /
   \   !    /
    \  !   /
     \ !  /
      \! /
       o
       !
       !
```

then in the tree file it is represented by the following sequence of printable characters, starting at the beginning of the file:

```
(B,(A,C,E),D);
```

The tree ends with a semicolon. Everything after the semicolon in the input file is ignored, including any other trees. The bottommost node in the tree is an interior node, not a tip. Interior nodes are represented by a pair of matched parentheses. Between them are representations of the nodes that are immediately descended from that node, separated by commas. In the above tree, the immediate descendants are B, another interior node, and D. The other interior node is represented by a pair of parentheses, enclosing representations of its immediate descendants, A, C, and E.
Tips are represented by their names. A name can be any string of printable characters except blanks, colons, semicolons, parentheses, and square brackets. In the programs a maximum of 30 characters are

allowed for names: this limit can easily be increased by recompiling the program and changing the CONSTant declaration for "nch" at the beginning of the program. Trees can have 300 nodes (including tips), this is controlled by the CONSTant "maxnodes" and can also be changed. Because you may want to include a blank in a name, it is assumed that an underscore character ("_") stands for a blank; any of these in a name will be converted to a blank when it is read in. Any name may also be empty: a tree like

```
 (,(,,),);
```

is allowed. Trees can be multifurcating at any level (while in many of the programs multifurcations of user-defined trees are not allowed or restricted to a trifurcation at the bottommost level, these programs do make any such restriction). Branch lengths can be incorporated into a tree by putting a real number, with or without decimal point, after a node and preceded by a comma. This represents the length of the branch immediately below that node. Thus the above tree might have lengths represented as:

```
 (B:6.0,(A:5.0,C:3.0,E:4.0):5.0,D:11.0);
```

These programs will be able to make use of this information only if lengths exist for every branch, except the one at the bottom of the tree.
The tree starts on the first line of the file, and can continue to subsequent lines. It is best to proceed to a new line, if at all, immediately after a comma. Blanks can be inserted at any point except in the middle of a species name or a branch length. The above description is of a subset of the New Hampshire Standard. For example, interior nodes can have names in that standard, but if any are included the present programs will omit them.
To help you understand this tree representation, here are some trees in the above form:

```
 ((raccoon:19.19959,bear:6.80041):0.84600,((sea_lion:11.99700,seal:12.00300):7.573,
 ((monkey:100.85930,cat:47.14069):20.59201,
weasel:18.87953):2.09460):3.87382,dog:25.46154);

 (Bovine:0.69395,(Gibbon:0.36079,(Orang:0.33636,(Gorilla:0.17147,(Chimp:0.19268,
Human:0.11927):0.08386):0.06124):0.15057):0.54939,Mouse:1.21460);

 ((A,B),(C,D));
```

The New Hampshire Standard was adopted June 26, 1986 by an informal committee meeting during the Society for the Study of Evolution meetings in Durham, New Hampshire and consisting of James Archie, William H.E. Day, Wayne Maddison, Christopher Meacham, F. James Rohlf, David Swofford, and myself. "

3.2. The New Hampshire eXtended Format (NHX)

This document is available in the distribution and at:
http://phylogenomics.us/forester/NHX.html

NHX is based on the New Hampshire (NH) standard (also called "Newick tree format"). It has the following extensions (compared to NH as used in the PHYLIP package):

- it introduces tags to associate various data fields with a node of a phylogenetic tree
- both internal and external nodes can be tagged

- number of children per node is at least two (allows polytomous trees)
- the tree is assumed to be rooted if the deepest node is a bifurcation
- the order of the tags does not matter, with the exception that the sequence name must be first (if assigned)
- the length of all character string based data is unlimited (name, species, EC number)
- Comments between '[' and ']' are removed (unless the opening bracket is followed by "&&NHX")

In order to remain compatible with the NEXUS format, all fields except sequence name and branch length (in other words, all fields eXtending NH) must be wrapped by "[&&NHX" and "]". E.g. "`ADH1:0.11[&&NHX:S=human:E=1.1.1.1]`".

In contrast to its name, NHX also has restrictions compared to Felsenstein's definition of the NH format: "Empty" nodes are not allowed (e.g. "(,(,),)" is not acceptable).
The following characters can not be part of names: '(' ')' '[' ']' ',' ':' as well as white spaces.

The tags are as follows.

TAG	**VALUE**	**MEANING**
no tag	String	sequence name of this node (MUST BE FIRST, IF ASSIGNED)
:	double	branch length to parent node (MUST BE SECOND, IF ASSIGNED)
:B=	double	bootstrap value at this node (does not apply to external nodes)
:C=	r.g.b	parent branch color (e.g. C=0.255.0 creates a green branch)
:Co=	'Y' or 'N'	collapse this node when drawing the tree (default is not to collapse)
:D=	'Y' or 'N'	'Y' if this node represents a duplication event – 'N' if this node represents a speciation event (does not apply to ext nodes)
:E=	String	EC number at this node (http://www.chem.qmul.ac.uk/iubmb/enzyme/)
:L=	float	log likelihood value on parent branch
:O=	integer	orthologous to this external node
:S=	String	species name of the species/phylum at this node
:SN=	integer	"subtree neighbor"
:SO=	integer	"super orthologous" (no duplications on paths) to this external node
:Sw=	'Y' or 'N'	placing a subtree on the parent branch of this node makes the tree significantly worse according to Kishino/Hasegawa test (or similar)
:T=	integer	NCBI taxonomy ID of the species/phylum at this node
:W=	integer	width of parent branch
:XB=	String	custom data associated with a branch
:XN=	String	custom data associated with a node

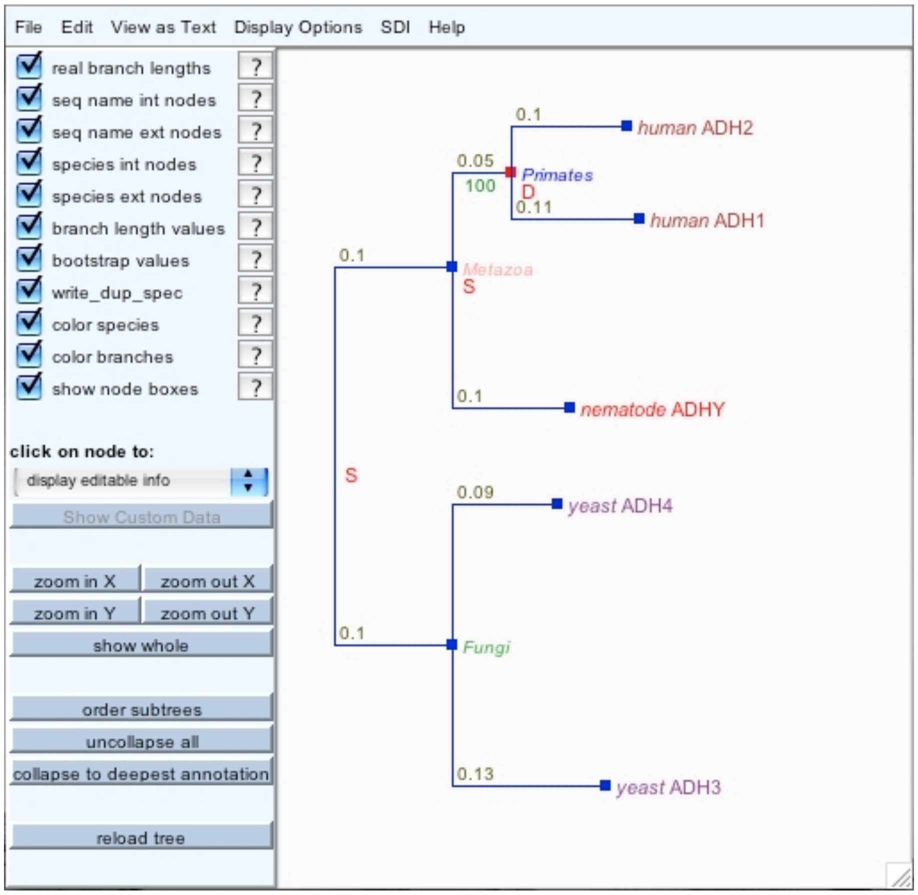
In Java, the data types are defined as follows:

String:	character string of arbitrary length
double:	64bit signed floating point number
float:	32bit signed floating point number
integer:	32bit signed integer number

An example of a (rooted) Tree in NHX:

```
(((ADH2:0.1[&&NHX:S=human:E=1.1.1.1],ADH1:0.11[&&NHX:S=human:E=1.
1.1.1]):0.05[&&NHX:S=Primates:E=1.1.1.1:D=Y:B=100],ADHY:0.1[&&NHX
:S=nematode:E=1.1.1.1],ADHX:0.12[&&NHX:S=insect:E=1.1.1.1]):0.1[&
&NHX:S=Metazoa:E=1.1.1.1:D=N],(ADH4:0.09[&&NHX:S=yeast:E=1.1.1.1]
,ADH3:0.13[&&NHX:S=yeast:E=1.1.1.1],ADH2:0.12[&&NHX:S=yeast:E=1.1
.1.1],ADH1:0.11[&&NHX:S=yeast:E=1.1.1.1]):0.1[&&NHX:S=Fungi])[&&N
HX:E=1.1.1.1:D=N];
```

This tree displayed in ATV:



3.2.1 Custom Tags

Custom tags take the form:

```
:X[N|B]=[S|D|L|B|U]=<tag>=<value>[=<unit>]
```

where

XN – custom data is for a node
XB – custom data is for a branch

and the data types are:

S – string
D – double
L – long
B – Boolean
U – URL

For example:

```
[&&NHX:S=alligator:XN=S=status=certain:XN=D=genome=4.3=MB]
```

3.3 PhyloXML

PhyloXML is an XML language for describing trees. See the document PhyloXML.pdf included with this distribution and http://www.phyloxml.org/ for more information.

4. Using ATV

In the following the term "ATV" refers both to the applet and application unless noted otherwise.

4.1. Menus

All of the ATV menu items are discussed below. Note that some menu items are only available in the application form of ATV, and some are only available for the applet form.

File Menu:

Reload tree – Restore the displayed tree to its original form. Note that this does not literally reload the file.

Open tree [URL] – Open a tree file in NH, NHX, or PhyloXML format from a file or URL. **Note**: if ATV is running as an applet, trees can only be opened as URLs.

Save [As] -- Save the displayed tree (which may have been edited).

Print [Options] – Print the displayed tree. Options include printing in color or black and white, and scaling the image. **Note**: printing is only available when ATV id running as an application.

Close – Close ATV.

Edit Menu:

Remove root – Remove the root of the displayed tree.

Remove root and trifurcate – Remove the root of the displayed tree and trifurcate.

Search – Search for all occurrences of a string. This will match fragments as well as complete words. For example, "At3" will match all gene names in a tree that begin with "At3".

Select Sequences – Type or paste a list of sequence names (separated by commas, spaces, semi-colons, or new-lines) and they will be highlighted on the tree.

Unselect All – Remove the highlighting from all selected nodes and sequences.

View as Text Menu:

This menu displays the displayed tree as text in any of three formats. The text may then be copied to the clipboard.

View as PhyloXML – Display tree data as PhyloXML.

View as NH – Display tree data in NH format.

View as NHX – Display tree data in NHX format.

Display Options Menu:

Switch colors – Switch to a new color set for displaying the tree.

Tiny fonts – Use tiny fonts on the tree.

Small fonts – Use small fonts on the tree.

Medium fonts – use medium fonts on the tree (default).

Large fonts – use large fonts on the tree.

SDI Menu:

This menu allows the inference of species and duplication events on a gene tree (given a species tree). For more information see:

Zmasek C.M. and Eddy S.R. (2001). A simple algorithm to infer gene duplication and speciation events on a gene tree. Bioinformatics, 17, 821-828.
http://bioinformatics.oxfordjournals.org/cgi/content/abstract/17/9/821

Help Menu:

Help – display a popup with simplified UI help.

Show/Hide item help – shows or hides the '?' boxes next to each checkbox items in the side control panel.

About – displays information about the version of ATV.

4.2. The Control Panel and Configuration File

Most of the controls in the control panel are configurable. This allows you to remove unneeded controls or to define default settings so you don't have to reconfigure your tree display every time you load a new tree. A sample file is included in the distribution, **ATVconfig.conf**. See this file for more information.

4.2.1. Display Options

There are several check boxes that control the appearance of the displayed tree:
box nodes – checked: display boxes at each node; unchecked: don't display boxes.

branch length values – checked: display branch length values on the branches; unchecked: don't display branch length values.

bootstrap values – checked: display bootstrap values on the branches; unchecked: don't display bootsrap values.

color acc to species – if data contains species information (NHX tag S=string, PhyloXML element): checked: display sequence names in different colors for each species; unchecked: all sequence names are displayed in the same color.

color branches – if data contains branch color information (NHX tag C=r.g.b, PhyloXML element `<branch>`): checked: display branches with colors designated in the data; unchecked: all branches are the same color.

display orthology – if data contains information about orthology (NHX tag 0=int, PhyloXML element <sequence_relation>): checked: display orthology information; unchecked: don't.

display s-orthology – if data contains information about super orthologs (NHX tag SO=int, PhyloXML element <sequence_relation>): checked: display orthology information; unchecked: don't.

display sub-neighbors – if data contains information about subtree neighbors (NHX tag SN=int, PhyloXML element <sequence_relation>): checked: display subtree neighbors; unchecked: don't.

duplic vs spec – if data contains information about duplication vs speciation (NHX tag D=Y/N, PhyloXML element `<event>`): checked: show duplication and speciation; unchecked: don't.

ec ext nodes – if data contains EC numbers on external nodes (NHX tag E=string, PhyloXML element `<EC_number>`): checked: show EC numbers; unchecked: don't.

ec internal nodes – if data contains EC numbers on internal nodes (NHX tag E=string, PhyloXML element `<EC_number>`): checked: EC numbers; unchecked: don't.

real branch lengths – checked: show branch lengths in data; unchecked: branches are all uniform lengths.

seq name ext nodes – checked: display sequence names on the external nodes; unchecked: don't display sequence names.

seq name int nodes – checked: display sequence names on the internal nodes; unchecked: don't display internal sequence names.

species ext nodes – checked: display species names on the external nodes; unchecked: don't display species names.

species int nodes – checked: display species names on the internal nodes; unchecked: don't display species names.

width branches – if data contains branch width information (NHX tag W=int, PhyloXML element `<branch>`): checked: display branches with widths designated in the data; unchecked: all branches are the same width.

4.2.2. Custom Data

Your tree file may contain custom data (see NHX and PhyloXML descriptions). If so, the 'Show Custom Data' button will be active and the custom data fields will be displayed in the popup-menu that appears when you click the button. Use this menu to indicate which custom data fields to display on the tree.

4.2.3. The Click-to Options

Every node in the tree is "hot" and can be clicked on to generate some action. You have to select an action from the "click-to" menu. The default is usually 'edit info'. The contents of this list along with the default selection can be modified by editing the .conf file. Here are the click-to options:

edit info – pop up a box showing all of the editable information for the clicked node. See section 4.3. Editing Node Information.

collapse uncollapse – a toggle option that collapses the clicked node, or expands the node if it's been collapsed.

reroot – re-root the tree using the clicked node as the new root.

subtree – show only the subtree whose root is the clicked node.

swap – swap the branches from the clicked node.

display sequences – pop up a box listing all of the sequences on the terminal nodes of from the clicked node.

display glyph popup – pop up a box showing the glyphs for the selected subtree. See 3.6. Glyph Panels for more information

go to swiss prot – **applet-only option**; opens a new browser window displaying the corresponding SWISS-PROT entry if the sequence starts with a SWISS-PROT name opens a new window. (http://www.expasy.org/sprot/)

display node popup – **applet-only option**; display a webpage with more information about the sequences in the clicked subtree. See section 5.2. Display Information About a List of Sequences for more information.

4.2.4. Zooming and Whole Tree Manipulation

The set of zoom buttons allows you to adjust the size of the tree in the X and Y dimensions independently, or to fit the whole tree in the current window.

At the bottom of the control panel is a set of buttons for quick, whole-tree manipulation:

order subtrees – swap the children of each node according to the number of external nodes.

uncollapse all – uncollapse all collapsed nodes.

collapse to deepest annotaion -- collapse the deepest nodes annotated with either a species or a sequence name.

reload tree – removes all edits and return to the original tree. Note that this does not literally re-read the file.

4.3. Editing Node Data

Data fields can be set for individual sequences, or for nodes on the tree. The following is a list of the fields that can be set in ATV version 3.01. New fields may be added in later versions. See the README file included in the distribution for information specific to the version.

seq name	– the sequence name
species	– the species name
taxonomy ID	– taxonomy ID
EC number	– the Enzyme Commission number (http://www.chem.qmul.ac.uk/iubmb/enzyme/)
GO term (term or ID)	– The Gene Ontology term or ID associated with this sequence or node (http://www.geneontology.org/)
distance to parent	– the length of the branch back to this node's parent
color of parent branch	– the color the parent branch, coded as red value, green value, blue value, each with a range from 0 (no color) to 255 (max color). Default color is determined by the color set. See the Display Options Menu for more information about color sets.
width of parent branch	– the width of the parent branch. The default width is 1 pixel.

duplication/speciation/not assigned – these radio buttons appear for internal nodes only; choose one to indicate the reason for the branch split.

4.4. Searching and Selecting Nodes

There are a number of ways that nodes can be highlighted (selected) and some operations can be carried out on a set of selected nodes. For example, a set of nodes can all be edited as a group (see section 4.3. Editing Node Data) though fewer data fields are valid than for a single node.

Search: The menu item Edit/Search will match text either exactly, or will match anything text on the tree which contains the indicated text. For example, "At1" will match all sequence names that begin with "At1." Once one or more sequence names are highlighted, they are also marked as selected and any operation that is valid on a set of nodes (e.g. edit info) can be carried out.

Select Sequences: The menu item Edit/Select Sequences takes a list of one or more sequences and highlights them on the tree. Again, once the sequences are highlighted, they are also selected as a group.

Shift-click: A set of nodes can also be selected using the standard shift-click method.

Unselect: A set of nodes can be unselected with the Edit/Unselect All menu item, or by clicking anywhere on the tree panel.

4.5. Glyph Panels

A “glyph panel” is an extra panel that appears left of the tree, associating each terminal node in the tree with an image or graph. For example, the image could show a simplified expression graph, the sequence location on a chromosome, intron/exon structure, and so on. Future versions of ATV will provide better support for these. As of version 3.1, glyph panels displaying chromosome location and a simple tissue expression graph have been implemented but remain prototypical and are not generally useful. See http://www.phylogenomics.us/atv/examples/index.html for examples of these two glyph panels. See the README for the current distribution for the most up-to-date information about glyph panels.

4.6. Manipulating the tree image beyond the capabilities of ATV

An example of a easy and powerful approach to manipulate the tree image beyond the capabilities of ATV is as follows (requires Adobe® Acrobat® and Adobe® Illustrator®). 1. Save the tree image to a PDF file by printing to Acrobat® PDFWriter instead of to a printer. 2. Use Illustrator® to manipulate the PDF image of the tree.

4.7. A Few Definitions

Super-ortholog: Given a rooted gene tree with duplication or speciation assigned to each of its internal nodes, two sequences are super-orthologous if and only if each internal node on their connecting path represents a speciation event.

subtree-neighbor: Given a completely binary and rooted gene tree, the k-subtree-neighbors of a sequence q are defined as all sequences derived from the k-level parent node of q, except q itself (the level of q itself is 0, q's parent is 1, and so forth).

5. Extending ATV’s Capabilities

ATV can be extended in two ways: by writing your own Java application around the ATV code, and by creating custom web pages to display additional information about individual sequences or lists of sequences.

5.1. Using ATV in Your Java Program

Here is an example of how to use packages “forester.tree” and “forester.atv” to display a phylogenetic tree in a JFrame. For more examples please refer to the source code files “ATVapp.java”, “ATVapp_awt.java”, “ATVjapplet.java”, and “ATVapplet.java”.
For more information about FORESTER and ATV please refer to the API specification in directory “forester_API_specification.”

IMPORTANT NOTE: As of the writing of this document, the API documentation has not been updated for ATV3. The ATV code has changed significantly so the old documentation will be of no use. See the README file with the current distribution to see if the documentation has been updated.

```
    // Imports packages from forester framework.
    import java.io.*;
    import org.forester.atv.*;
    import org.forester.phylogeny.*;

    public class myClass {

        public void myMethod() {

            Phylogeny tree = null;
            String config_file = “AVTConfig.conf”;

            try {
                File f = new File( “testtree.nhx” );
                // Reads in a tree (in NH or NHX format) from a file (“testtree.nhx”).
                tree = Helper.readNHtree( f, false );
            }
            catch ( Exception e ) {
                System.err.println( “Could not read tree. Terminating.” );
                System.exit( -1 );
            }

            // Displays the tree in a JFrame.
            ATVframe atvframe = new ATVframe( tree, config_tree );
        }
    }
```

5.2. Display Information About a Sequence

A relatively simple way to display additional information about a sequence is to link individual sequence names to a web page, for example, the GenBank (http://www.ncbi.nlm.nih.gov/Genbank/index.html) record for that sequence, or perhaps to a suitable GBrowse (http://www.gmod.org/?q=node/71) installation.

Note: this functionality is only available for applets.

To attach an external web page to individual sequences:

1. Prepare a web page that will accept a sequence name as a URL parameter and display information about that sequence.

2. Change one line in the .conf file:

   ```
   sequence_popup_URL:
   ```

   ```
   sequence_popup_URL: URL-for-sequence-info?seq=
   ```

The sequence name will be appended to the end of this string before the page from step 1. is requested in a new browser window.

5.3. Display Information About a List of Sequences

Similarly, it is possible to send a list of sequence names to a web page. This could be used, for example, to display all the sequences in a branch on a genome map.

Note: this functionality is only available for applets.

To attach an external web page to a set of sequences:

1. Prepare a web page that will accept a comma-separated list of sequence name as a URL parameter and display information about those sequences.

2. Change this line in the .conf file:

```
click_to: display_node_popup  nodisplay
```

to

```
click_to: display_node_popup  display
```

so that the option will appear in the click-to list.

3. Also in the .conf file, indicate the URL to go to:

```
sequence_popup_URL: URL-for-sequence-info?seqs=
```

A comma-separated list of sequence names will be appended to end of this string before the page from step one is requested in a new browser window.

4. If you want this to be the default click-to behavior, then change this line in the .conf file:

```
default_click_to: edit_info
```

to

```
default_click_to: display_node_popup
```

6. Obtaining Java

To download the current version of Java:
http://java.sun.com/

7. Sources for More Information

Java
Java: http://java.sun.com/
Jar files: http://java.sun.com/j2se/1.5.0/docs/guide/jar/index.html
Swing: http://java.sun.com/docs/books/tutorial/uiswing/14start/index.html

Tools
ATV: http://www.phylogenomics.us/atv
PHYLIP: http://evolution.genetics.washington.edu/phylip.html
SDI: http://www.genetics.wustl.edu/eddy/forester/SDI_manual.html

Tree formats
PhyloXML: http://www.phyloxml.org/
NH: http://evolution.genetics.washington.edu/phylip/newicktree.html
NHX: http://www.genetics.wustl.edu/eddy/forester/NHX.html

NCBI taxonomy: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Taxonomy

8. References

ATV: Zmasek C. M. and Eddy S. R. (2001) ATV: display and manipulation of annotated phylogenetic trees. *Bioinformatics*, 17, 383-384.
SDI: Zmasek C. M. and Eddy S. R. (2001) A simple algorithm to infer gene duplication and speciation events on a gene tree. *Bioinformatics*, in press.

9. Copyright Information