ARAUCARIA

VERSION 3.1

User Manual

UNIVERSITY OF DUNDEE

Chris Reed & Glenn Rowe
School of Computing, University of Dundee

May 2006

The authors gratefully acknowledge the generous support for the work provided by the School of Computing at the University of Dundee, Dundee DD1 4HN Scotland, UK.

Contents

1. Introduction

Welcome to Araucaria, a software tool for analysing and diagramming arguments.

Araucaria can be used

- in preparing teaching materials in critical thinking, informal logic and argumentation theory
- in the classroom, either for instructor or student use
- for preparing online resources
- for working with argumentation schemes
- in designing examples for academic work
- in exchanging examples and problems in a common, open format (AML)
- for the reuse and sharing of material between individuals and sites
- for building and accessing a large, online repository of argumentation (AraucariaDB)

This manual is written to be useful to students, educators, and researchers.

Araucaria is free software developed (released under the GNU Public License) at the University of Dundee by Glenn Rowe and Chris Reed. Queries and comments concerning Araucaria should be directed to `araucaria@computing.dundee.ac.uk`.

Publications describing Araucaria, its aims and uses are available at the Araucaria web page, `araucaria.computing.dundee.ac.uk`. If you find Araucaria useful in your academic work, please include a reference to

> Reed, C.A. & Rowe, G.W.A. (2004) "Araucaria: Software for Argument Analysis, Diagramming and Representation", *International Journal of AI Tools* **14** (3-4) pp961-980

2. Getting Started

If you have not yet installed Araucaria, access the CD and view the file `index.html`.

You will be guided through the installation process. Further details of installation are given in sections 15 and 16. Once you have installed the software, you can run Araucaria by selecting Araucaria from the Start menu or double clicking `Araucaria.exe` (Windows) or running `./Araucaria.sh` (Mac/Linux/Solaris) in the Araucaria installed directory.

When you run Araucaria, you will see the Araucaria Main Window:

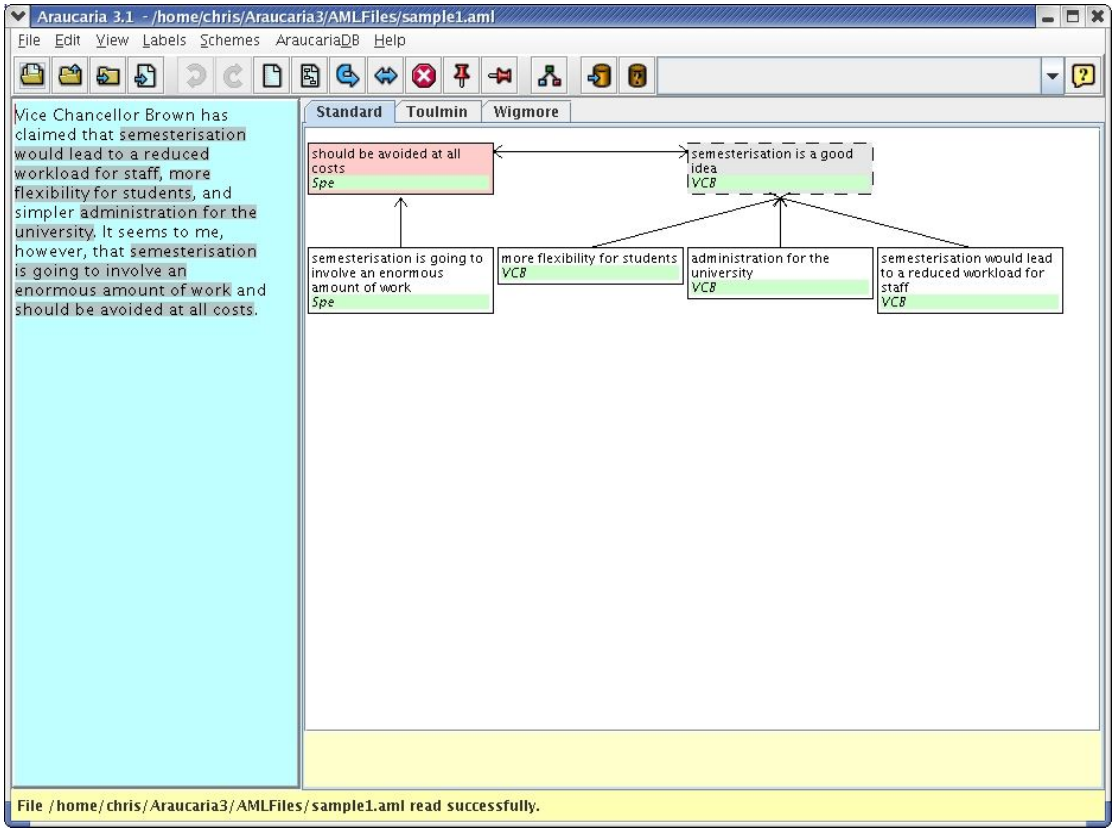

Figure 1. Araucaria Main Window

Araucaria is primarily a tool for **analysing** existing arguments. The first step is therefore to load the text of an argument.

The distribution CD includes some sample arguments, so try loading `TextFiles/sample1.txt` by clicking File - Open text file, and then selecting the file `sample1.txt` from the `TextFiles` directory.

The text now appears in the blue, left-hand pane. You can now start identifying premises and conclusions in this argument. So, for example, to identify *semesterisation is going to involve an enormous amount of work* as a premise, select that text by holding down the mouse button and dragging from the *s* of *semesterisation* to the *k* of *work*. Once the text is selected (indicated by a grey box around the text), release the mouse button. You can then add the selected premise to the diagram. To do this, simply click once on the white, right-hand diagram area.

You should now see a **node** at the bottom of the diagram area - a circle with the letter A inside it.

Now try selecting a conclusion in the text - maybe the phrase *should be avoided at all costs*. Drag and click as before. You should now have a second node, with the letter B.

To construct an argument diagram showing that A supports B, simply click on A, hold the mouse button down, and drag the dashed line onto B. Release the button and you should have a diagram with A supporting B.

Now that you can add nodes to the diagram and then connect them together, you can move on to learn about

- drawing argument diagrams in which several premises support a conclusion either independently or in a linked structure
- how premises can support premises
- how to delete components from the diagram
- how to construct refutations
- how to insert missing premises
- how to save your analysis and the associated diagram

3. Constructing the Diagram

Using the select-and-click technique for adding nodes to the diagram, you should be able to include all the propositions expressed in the text of an argument in the diagram. Parts of the diagram, including nodes and the arrows between nodes are called **diagram components**.

Selecting Diagram Components

To **select** a node or arrow in the diagram, simply click on it once. If you select a node, a dark box or ring will appear around the node, and the text associated with the node will appear in the yellow **message area** at the bottom of the window. If you select an arrow between nodes, the arrow will become blacker, and text that corresponds to the nodes at either end of the arrow will be displayed.

If you right-click on any component in the diagram (Mac users may not be able to do this), a special **popup menu** will appear with various options for that component.

To select more than one diagram component at once, first select a single component, then hold down the Shift key and select further components. All components will then be marked by a heavy black line.

To deselect one or more components, simply click anywhere on the diagram area.

Deleting Diagram Components

To delete anything from the diagram, simply select the component to be deleted, and then do one of the following:

- Hit the 'Del' key
- Click the  icon on the toolbar
- On the menu, click 'Edit - Delete'

Connecting and Unconnecting Nodes

To support node X with node Y, click on Y and hold the mouse button down while dragging the dotted line to X.

Any node that is not already supporting some other node can be used as a support.

No one node may support more than one other node. Araucaria does not currently support a single premise supporting multiple conclusions (known as **divergent** argumentation).

To remove the support between two nodes, select the support arrow, and delete it. The diagram will rearrange itself automatically to accommodate the changes you make.

Linking and Unlinking Premises

Some arguments involve premises which support a conclusion independently: these are usually called **convergent** arguments. In contrast, other arguments involve premises which have to be taken together in supporting a conclusion: these are usually called **linked** arguments.

Premises can be linked together in Araucaria. Select all of the premises to be linked (they must all be supporting a single conclusion) and then do one of the following

- Hit ’Ctrl-L’
- Right-click on one of the selected premises and choose 'Link' from the pop-up menu
- Click the  icon on the toolbar
- On the menu, select ‘Edit - Link Statements’

To unlink several linked premises, select any or all of the linked premises or arrows and then do one of the following:

- Hit ’Ctrl-U’
- Right-click on one of the selected premises and choose 'Unlink' from the pop-up menu
- Click the  icon on the toolbar
- On the menu, select ‘Edit - Unlink statements’

Missing Premises

Some arguments involve premises that are left implicit or tacit. Such arguments are usually called **enthymemes**. If an argument you are analysing involves an enthymeme, Araucaria allows you to insert missing premises.

To add a missing premise, do one of the following:

- Hit 'Ctrl-M'
- Click the  icon on the toolbar
- On the menu, select 'Edit - Missing premise'

You will then be presented with a dialog box into which you should type the text to be associated with the missing premise. Click OK when done.

A new node will then appear on the diagram, which can be connected in the usual way. To delete a missing premise, select it and delete it as usual. Notice that missing premise nodes are displayed in grey and have dashed borders.

The text of a missing premise node can be altered after it has been added to the diagram. To alter the text of a missing premise node,

- Select and right-click the node, and select 'Edit Text' from the pop-up menu.

Refutations

In some situations, your analysis may require the notion of **refutation**. Araucaria employs a specific meaning for refutation, which is close to the logical idea of **negation**.

If you want to make node X a refutation of node Y, first join X in to the diagram by making X support Y (dragging a line from X to Y). Then select X, and do one of the following:

- Hit 'Ctrl-R'
- Click the  icon on the toolbar
- On the menu, select 'Edit - Refutation'

X can work as the refutation of Y if and only if X expresses the converse of Y. Perhaps the easiest way of understanding this is to see if X is equivalent to saying "it is not the case that Y". Notice that this means that one node can have a maximum of **one** refutation.

To build an analysis in which there are many 'counterarguments' (or, more specifically, for which there are multiple rebuttals) introduce a refutation node and then drag supports to it as usual. (In contrast, introducing undercutting arguments means adding refutations to premises. The terms undercut and rebut are from Toulmin, *The Uses of Argument*, CUP, 1958. See also section 6).

Refutation nodes are shown in pink in the diagram. Nodes that are both refutations and enthymemes are shown shaded in pink and grey with a dashed border.

Inverting the Diagram

Argumentation textbooks sometimes draw arguments with the conclusion at the top, and sometimes with the conclusion at the bottom. Araucaria can draw diagrams either way up. To flip the diagram over, do one of the following:

- Hit 'Ctrl-F'
- Click the  icon on the toolbar
- On the menu, select 'Edit - Flip diagram'

Undoing Changes

Any change to the argument diagram that you are working with can be undone. To undo a change, do one of the following:

- Hit 'Ctrl-Z'
- Click the  icon on the toolbar
- On the menu, select 'Edit - Undo'

If you change your mind and want to redo the change that you have just undone, do one of the following:

- Hit 'Ctrl-Y'
- Click the  icon on the toolbar
- On the menu, select 'Edit - Redo'

Starting Again

If you want to clear your analysis and start again on the same text, do one of the following:

- Hit ’Ctrl-C’
- Click the  icon on the toolbar
- On the menu, select ‘Edit - Clear diagram’

Closing and Exiting

To close your files

- Hit ’Ctrl-N’
- On the menu, select 'File - Close all'

You will be prompted if there is unsaved work that you might lose.

To exit Araucaria completely,

- On the menu, select 'File - Exit'

You will be prompted if there is unsaved work that you might lose.

Getting Help

Although this manual offers the most comprehensive guide to Araucaria, summary online information is available by doing one of the following:

- Hit ’F1’
- Click the  icon on the toolbar
- On the menu, select ‘Help - Help’

Saving

You can save your analysis as an argument in **AML** (see section 12 for more details of the Argument Markup Language used by Araucaria).

To save an argument, do one of the following:

- Hit ‘Ctrl-S’
- Click the  icon on the toolbar
- On the menu, select ‘File - Save Argument’

Once an argument analysis is saved in AML, you can reload it into Araucaria (the text of an argument is saved along with the analysis and diagram). You can also use it in a number of other applications, details of which are available from the research team at Dundee - email `araucaria@computing.dundee.ac.uk`.

Arguments are saved with a `.aml` extension.

To give an analysis a new filename, use ‘File – Save Argument As’

You might also want to save the diagram itself, for later inclusion into a report or paper. To save the diagram currently visible in Araucaria, do one of the following:

- Hit ’Ctrl-D’
- Click the  icon on the toolbar
- On the menu, select ‘File - Save Diagram’

Araucaria can save images as either JPEG files (which are more compact) or TIFF files (which are higher quality, and are often used by publishers). Images of diagrams are saved with either a `.jpg` or a `.tif` extension.

Zooming In

With small analyses, the default view is probably most convenient, where you see the full text of every node displayed inside that node in the diagram. If the diagram becomes larger than the display window, you can use the scrollbars that appear to move the diagram around.

For larger analyses, it may be useful to be able to get a bird's eye view of the argument as a whole. To do this, you can select one of the other views.

Any Araucaria analysis can be viewed in one of three ways:

- **Full Text** in which the full text of every node is displayed, making for large diagrams
- **Full Size** in which each node is represented by an ID label (A, B, C, ...), but the image is guaranteed to remain clear with no overlapping components – in very large analyses, this can mean that the diagram will still not fit in the window
- **Scaled** in which each node is represented by an ID label (A, B, C, ...), and the image is guaranteed to fit in the display window – in very large analyses, this can mean that some components may be slightly overlapping

To change the current view, do one of the following

- Hit Ctrl and the minus ('-') key to zoom out, or Ctrl and the equals ('=') key to zoom in.
- On the menu, select 'View – Zoom >' and then one of 'Full Text', 'Full Size' or 'Scaled'

Note that when you save a diagram, Araucaria always saves whichever view you currently have open.

To view the text of a node in Full Size of Scaled views, do one of the following:

- Single click on the node, and the text will appear in the message area
- Right-click in the node, and select 'Show Text' from the popup menu to temporarily display the text in the diagram area

Collapsing

Full Text mode is useful for understanding an argument, but quickly fills the screen. To save space, you can **collapse** parts of the diagram. By collapsing all the argumentation in support of a particular node, you can temporarily hide all that supporting argumentation.

To collapse the support for a node, do one of the following:

- Double click on the node
- Select the node, right-click, and select ‘Collapse’ from the popup menu

When a node has collapsed supporting argumentation hidden beneath it, a small blue cross will be visible inside the node (in Full Text view) or just by the node (in the other views). To restore the hidden argument components, do one of the following:

- Double click on the node
- Select the node, right-click, and select ‘Expand’ from the popup menu

4. Using Labels

Araucaria supports labels that relate to two further aspects of argument: ownership and evaluation. Both individual claims (the nodes representing premises, conclusions and refutations) and also the supports between them (the arrows in the diagram) can be labelled to indicate

(i) which speaker is associated with them and

(ii) how the analyst has evaluated them.

This section describes how to include ownership and evaluation labels in your analyses.

Ascribing Ownership

Sometimes a single text may represent the arguments of more than one point of view - the text may be a summary, or it may be an argument which uses or attacks the arguments of others. Araucaria can diagram such arguments by marking who **owns** each claim (and of course, sometimes, many different views will agree on a single claim).

Each premise, conclusion, and refutation - every node in the diagram - can be **ascribed** to one or more **owners**. After selecting one or more nodes in the diagram, you can change the ownership information of those nodes by doing one of the following:

- Hit 'Ctrl-W'
- Right click on one of the nodes or arrows and select 'Modify evaluation' from the pop-up menu
- On the menu, select 'Labels - Modify Ownership'

This opens the ownership window:

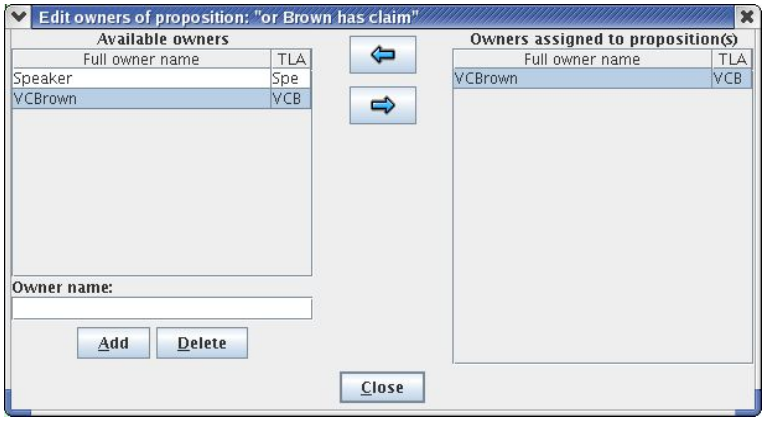

Figure 2. The Edit Ownership Window

To create a new owner (a new protagonist in the argument, if you like), enter the name of the owner in the **Owner name** box, and then click Add. The new name will be added to the list on the left. This list is a complete list of all owners currently recognised for this argument. A three letter acronym (TLA) is produced, and is used to label nodes in the diagram.

To ascribe ownership of the selected diagram components, select one or more owner names by clicking on them (hold down the Ctrl key to select several owner names). Then click the  icon. (If the  and  are not visible, make sure you have selected a node.)

To remove owners from those ascribed to the selected diagram components, simply click on them in the right-hand pane, and click the  icon.

To delete owners from the list available for the current argument, select one or more owners from the list on the left, and click the Delete button. Notice that this will remove the selected owner(s) from all throughout the argument, deleting all ascriptions for that owner.

Notice that a proposition cannot be ascribed to an owner until after it has been connected to the diagram - i.e. not immediately after the proposition node is created by dragging over text and dropping. Ownership can only be ascribed to nodes, and not to arrows between nodes.

Displaying Ownership

You can turn off the extra information in the diagram relating to ownership of propositions. To turn off ownership information in the image, do one of the following:

- Hit ’Ctrl-H’
- On the menu, select ‘Labels - Hide owners’

Turning off the information does not delete it. You can turn the ownership information in the diagram back on again by doing one of the following:

- Hit ’Ctrl-H’
- On the menu, select ‘Labels - Show owners’

(Note that being able to store information about owners does not mean that Araucaria can handle dialogue transcripts. That is an area of current work at Dundee.)

Ascribing an Evaluation

Sometimes the analysis of an argument may involve making judgements about the claims or about the relationships between claims. You may wish to indicate whether an argument is 'good' or 'bad', or 'stong' or 'weak'. You may wish to assign to claims particular values between 0 and 1 corresponding to probabilities. You may wish to mark the components of an argument according to some many-valued logic (with, say '+', '-', and '?').

Araucaria can support such analyses by marking **evaluations** on each node and arrow. The labels used to indicate evaluations are user-defined, so you can employ whatever set best meets your needs.

To assign an evaluation, select one or more nodes or arrows in the diagram and do one of the following:

- Hit ’Ctrl-K’
- Right click on one of the nodes or arrows and select 'Modify evaluation' from the pop-up menu
- On the menu, select ‘Labels - Modify evaluation’

This opens the evaluations window:

Figure 3. The Edit Evaluation Window

Enter the evaluation label you want to assign to the selected node(s) and click "Add evaluation".

If you have already entered evaluations elsewhere in your argument analysis, you can use the pull-down list box to select one to use again. Or you can just type it in again.

You can only assign a single evaluation to a given node in the diagram.

To delete an evaluation from one or more nodes in the diagram, select the node(s), open the evaluation window, and click "Delete evaluation".

Displaying Evaluations

You can turn off the extra information in the diagram relating to evaluations. To turn off evaluation information in the image, do one of the following:

- Hit ’Ctrl-G’
- On the menu, select ‘Labels - Hide evaluations’

Turning off the information does not delete it. You can turn the ownership information in the diagram back on again by doing one of the following:

- Hit ’Ctrl-G’
- On the menu, select ‘Labels - Show evaluations’

Changing the Node ID

By default, nodes are labelled with a single letter of the alphabet, in alphabetical order (after 52 nodes are identified, Araucaria uses two letters). You may wish to change this.

To change the Node ID of a particular node in the diagram,

- Select the node, right-click on it, and select 'Edit ID' from the pop-up menu

You can then enter a new Node ID. A Node ID can be up to two characters long, and is case sensitive. (For more details, see section 7).

Node IDs are only visible in Full Size and Scaled views.

5. Working with Schemes

Argumentation schemes offer a means of characterising stereotypical non-deductive patterns of reasoning. Identifying such schemes may make your analyses richer, and may be useful in teaching argumentation theory and critical thinking.

Different researchers have proposed different ways of arranging and classifying argumentation schemes.

Araucaria supports the use of argumentation schemes in a given analysis, and also offers the flexibility to use alternative sets of argumentation schemes, and even enables you to design your own argumentation schemes.

The distribution CD includes several sets of argumentation schemes (or **schemesets**):

- A set that corresponds to the schemes proposed by D. N. Walton in his book *Argumentation Schemes for Presumptive Reasoning* (LEA, 1996). This set includes many common schemes, and can be found in the file `walton.scm`
- A set that corresponds to reasoning patterns in J. L. Pollock's book *Cognitive Carpentry* (MIT Press, 1995). This is a concise set, and can be found in the file `pollock.scm`
- A set that corresponds to the schemes discussed by W. Grennan in his book *Informal Logic* (McGill-Queens U. Press, 1997). This can be found in the file `grennan.scm`
- A set that corresponds to the recent attempts at an exhaustive taxonomy of schemes at U. Dundee. This extensive set can be found in the file `dundee.scm`
- A set that corresponds to most of the rhetorical patterns discussed by Perelman and Olbrechts-Tyteca in their book *The New Rhetoric* (Notre Dame Press, 1969).This set can be found in the file `perelman.scm`

Further schemesets are being made available on the project website, at `http://araucaria.computing.dundee.ac.uk/`

If you have any questions about the use of argumentation schemes in Araucaria, or would like to submit a schemeset for others to use, please email the project's authors, at `araucaria@computing.dundee.ac.uk`.

Selecting an Argumentation Scheme

If your text includes an example or instance of an argumentation scheme, it should have components corresponding to the conclusion and at least some of the premises of the scheme.

First, diagram the argument as usual (see section 3). Then, to associate an argumentation scheme with a part of the argument, select the arrow or arrows which are included in the scheme. Finally, select a scheme by doing one of the following

- Hit 'Ctrl-E'
- Click the  icon on the toolbar
- On the menu, select 'Schemes - Select'

Araucaria will then display the scheme window:

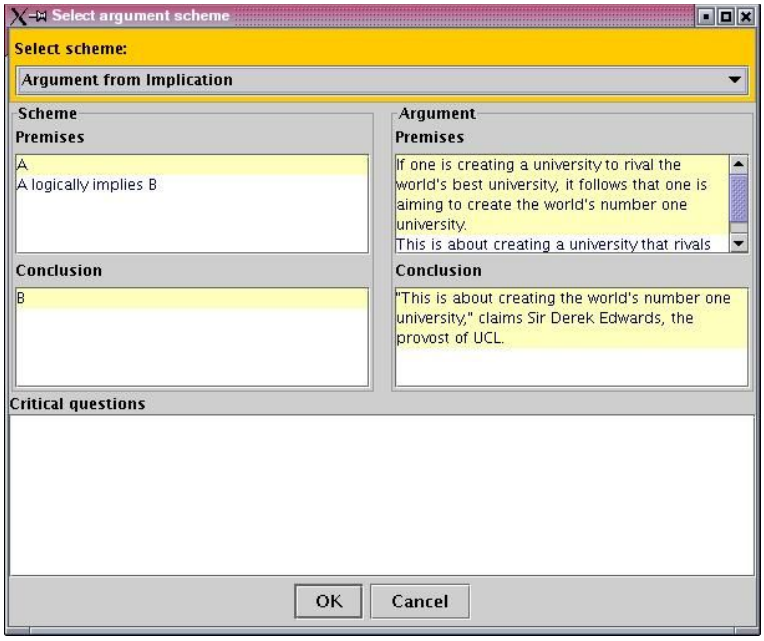

Figure 4. The Scheme Window

The top pull-down list box allows you to select the appropriate argumentation scheme.

The remaining text areas are for information, and may be particularly useful for students. The left-hand Premises and Conclusion boxes describe the generic form of the argumentation scheme. The right-hand Premises and Conclusion boxes describe the matching components from the text of the argument (though not necessarily in the same order). This may be helpful to students in identifying missing premises.

The final box lists the critical questions associated with the scheme. Again, this may be useful to students in evaluating the argument at hand.

To confirm that the currently selected diagram components form an instance of the selected argumentation scheme, click ‘OK’. The diagram will now be updated to show a coloured area demarcating the scheme.

An argument analysis in Araucaria

- may include several schemes
- may include multiple instances of the same scheme
- may include a node functioning in more than one scheme (e.g. the conclusion of an instance of one argumentation scheme may function as a premise in another) - argumentation schemes may thus ‘overlap’.

You can select an argumentation scheme in the diagram simply by clicking on it (the message area will then display the name of the scheme).

To remove a scheme from an analysis, simply select it by clicking on it, and thendo one of the following, as usual:

- Hit the ’Del’ key
- Click the  icon on the toolbar
- On the menu, click ‘Edit - Delete selected’

When deleting, be careful to select the scheme (by clicking in the coloured area) rather than a node or a support arrow.

If you have associated a particular scheme with part of your argument, but subsequently decide to change the association, you can do this by:

- Select the scheme you wish to change by clicking on it, and then right-click and select 'Edit scheme' from the pop-up menu

Adding and Editing Argumentation Schemes

The schemeset(s) currently available may not correspond to the argumentation schemes in your course examples, course notes, or adopted textbook, or may simply be inappropriate for the job at hand. In this case, Araucaria offers a simple interface for building your own argumentation schemes.

To add or edit an argumentation scheme, do one of the following:

- Hit 'Ctrl-I'
- On the menu, select 'Schemes - Add/Edit'

To add a new scheme, click 'New', or to edit an existing scheme, click 'Edit'. The argumentation scheme edit window then becomes available.

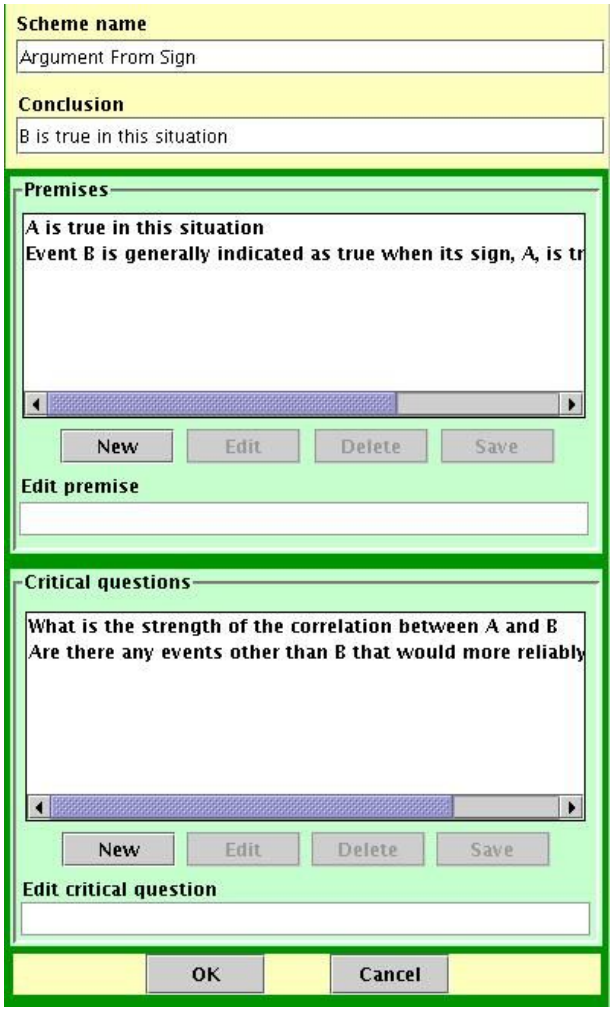

Figure 5. The Scheme Edit Window

First, enter the name you wish your new scheme to have (e.g.*Argument from Sign*), and then enter the general form of the conclusion of such an argument (e.g.*B is true in this situation*).

Next, add as many premises as you want. For each one, click ‘New’, then enter the general form of the premise (e.g.*A is true in this situation*), and hit ‘Enter’ after each premise. To edit or delete existing premises, select the premise you want to change, and then use the ‘Edit’ and ‘Delete’ buttons respectively.

Finally, add the critical questions to be associated with this scheme. Again, you can introduce as many as you want. Enter critical questions in the same way as premises.

When your new argumentation scheme is complete, click ‘OK’.

Editing schemes follows exactly the same process - you can change, remove, or add to the premises and critical questions, or can change the name or conclusion of any existing scheme.

Once you have completed your schemes, you will be able to select them during analysis as described above.

Opening and Saving Schemesets

If you make modifications to the default schemeset, you may wish to save your schemes for future use. To save a schemeset, do one of the following:

- Hit ’Ctrl-V’
- On the menu, select ‘Schemes - Save schemeset’

Schemesets are saved with a `.scm` extension.

To open a previously saved schemeset, do one of the following:

- Hit ’Ctrl-P’
- On the menu, select ‘Schemes - Open schemeset’

By default, no schemeset is loaded, and there are no schemes available. If you want to work with schemes, the easiest way to get started is to load a schemeset such as `walton.scm`, which is available on the distribution CD.

Notice that the schemeset used by a given argument is saved in the AML representation of that argument. This means that you can exchange AML arguments with colleagues without having to ensure that everyone has the same schemesets. You may however, wish to provide a common set of schemes for use in, for example, a classroom exercise, and it is to this scenario that schemesets are probably best suited.

6. Working in Other Styles: Toulmin

The conventional “box-and-arrow” type diagram is not the only way of diagramming arguments. As a result, Araucaria supports different diagramming **styles**. After version 3.0, the software supports not only “box-and-arrow” diagrams but also Toulmin diagrams, based on the approach described in S. E. Toulmin's book *The Uses of Argument* (CUP, 1958). Version 3.1 also introduced another style discussed in the next chapter. Future versions of Araucaria may support further styles.

To change the style of your analysis, do one of the following:

- Select a different tab from above the diagram pane – the tabs are labelled, ‘Standard’, ‘Toulmin’ or ‘Wigmore’
- Hit Ctrl and the comma key (‘,’) to cycle leftwards through the style tabs, or Ctrl and the period key (‘.’) to cycle rightwards through the style tabs
- On the menu, select ‘View – Style >’ and select one of ‘Standard’, ‘Toulmin’ or ‘Wigmore’ (notice that this menu shows options under current development; these are not selectable in version 3.1)

Changing the style of diagram does not affect the stored information about the analysis (technically, all styles are represented and saved using AML). So you can make a change in one style, then swap to another style and make another change, and so on. Most changes will be visible in other styles, and all changes are stored once made.

There are some differences between what can be achieved using the Standard style and what can be achieved using the Toulmin style. Some things can not be done using the Toulmin style. These include:

- Setting Ownership
- Using Schemes
- Building Linked arguments

These do not have a place in Toulmin diagrams. As a result, you will see that some toolbar icons and menu options become inactive under the Toulmin style.

Some things are done a little differently in the Toulmin style (evaluations and refutations, for example). But otherwise, all the basic mechanisms for Standard diagrams also work for Toulmin diagrams.

There are interesting and difficult problems in translating between styles – some of these form the topics of research papers at `http://araucaria.computing.dundee.ac.uk`. This section describes how to use the Toulmin style independently of the Standard style.

Constructing a Toulmin Diagram

Creating nodes, selecting diagram components, and deleting diagram components works exactly the same as for Standard style diagrams (see sections 2 – 3).

Connecting components is mostly the same. Toulmin diagrams involve nodes with different **roles**: Datum, Claim, Warrant, Backing, Rebuttal, and Qualifier. The different roles are identified by different coloured boxes, and, in Full Size and Scaled views, with different shapes as well. The different roles require slightly different rules for diagramming.

Dragging the dotted line from one node to another will create a Toulmin argument with a Datum and a Claim. To add a Warrant, click a node and drag onto an arrow between an existing Datum and Claim. To add a Backing, click and drag onto an existing Warrant.

To add a Qualifier, select the arrow between a Datum and a Claim, right-click, and from the popup menu, select 'Add/Edit data qualifier'. This opens the same evaluation label window as in the Standard style (section 4). Select or type in an evaluation and click OK.

Changing Roles

It is possible to create some components by **converting** them from others.

To add a Rebuttal, first drag the node onto the Datum – Claim arrow as a Warrant. Then do the following:

- Select the Warrant, right-click, and from the popup menu, select 'Toulmin role'. From the menu, select 'Convert to Rebuttal'.

To do the reverse and transform a Rebuttal into a Warrant, do the following:

- Select the Rebuttal, right-click, and from the popup menu, select 'Toulmin role'. From the menu, select 'Convert to Warrant'.

If you make a mistake you can undo, delete and redo work as in Standard style. But in addition, it is possible to swap nodes around.

To exchange a Datum and a Warrant, do one of the following:

- Select the Warrant, right-click, and from the popup menu, click‘Swap with Datum’
- Select the Warrant and then, using the Shift key, also select the Datum, right-click on Datum, and click ‘Swap with Warrant’
- If there is only one Warrant, you can also Select the Datum, right-click, and click ‘Swap with Warrant’

To exchange a Datum and a Rebuttal, do one of the following:

- Select the Rebuttal, right-click, and from the popup menu, click‘Swap with Datum’
- Select the Rebuttal and then, using the Shift key, also select the Datum, right-click on Datum, and click ‘Swap with Rebuttal’
- If there is only one Rebuttal, you can also Select the Datum, right-click, and click ‘Swap with Rebuttal’

To exchange a Rebuttal and Warrant, do one of the following:

- Select the Rebuttal and then, using the Shift key, also select the Warrant, right-click on Warrant, and click ‘Swap with Rebuttal’
- Select the Warrant and then, using the Shift key, also select the Rebuttal, right-click on Rebuttal, and click ‘Swap with Warrant’
- If there is only one Rebuttal, you can also select the Warrant, right-click, and from the popup menu, click ‘Swap with Rebuttal’
- If there is only one Warrant, you can also Select the Rebuttal, right-click, and click ‘Swap with Warrant’

Chaining Arguments

To glue together multiple arguments in the Toulmin style, simply drag and drop as usual. Every node in a Toulmin diagram (except Qualifiers) can act as a Claim in another argument. So A can be Datum for the Claim of B, and B can be Datum for the Claim of C.

If you chain together many arguments, you may sometimes want to support a Warrant with a single Backing, and at other times, to support a Warrant with a whole other argument. To do this, create a Backing as usual, then select that Backing, right click, and from the popup menu, select 'Toulmin role'. From the menu, click 'Convert to Datum', and the Backing will become a new Toulmin argument with the Warrant as its Claim. It is possible to reverse this process by selecting a Datum that supports a Claim that is functioning as a Warrant in another argument, and clicking 'Convert to Backing' from the popup menu.

Advanced Notes on Data

A Toulmin diagram always has a Datum. In Araucaria, if you delete the Datum (or if you transform it to something else, as described above) it will, if necessary, be replaced by a Datum "Placeholder", in a greyed box with dashed edges (like enthymemes) labelled with "?". This placeholder cannot be deleted (unless you break the argument back in to pieces). To insert a real Datum, add a new Warrant and then swap it with the placeholder Datum. Alternatively, if you want to use an enthymeme, simply right-click on the Datum, and select 'Edit text' as usual.

Advanced Notes on Warrants

It is possible to construct Toulmin diagrams that have more that one Warrant. Simply drag as many nodes onto the Datum – Claim arrow as you want.

For every Warrant, a new Qualifier can also be added. To do this, click on the arrow from the Warrant to the Datum – Claim arrow, right-click, and from the popup menu select 'Add/edit warrant qualifier'. The new Qualifier will appear on the Datum-Claim arrow as usual, but it will be immediately next to the Warrant to which it is attached.

Advanced Notes on Rebuttals

As with Warrants, it is possible to construct Toulmin diagrams that have more that one Rebuttal. Simply transform as many Warrants as you like into Rebuttals.

Again, as for Warrants, for every Rebuttal, a new Qualifier can also be added. To do this, click on the arrow from the Rebuttal to the Datum – Claim arrow, right-click, and from the popup menu select 'Add/edit rebuttal qualifier'. The new Qualifier will appear on the Datum-Claim arrow as usual, but it will be immediately next to the Rebuttal to which it is attached.

Swapping a Rebuttal to a Warrant and vice versa retains any attached Qualifier.

Technical Note. The way that Araucaria interprets Rebuttals is as refutations of implicit specialised Warrants. These implicit Warrants make little sense in the standard Toulmin diagram. Usually, therefore, they are hidden. There are occasions, however (particularly when automatically converting from other styles), when it is appropriate to show them. It is possible to force Araucaria to show an implicit Warrant by right-clicking on the Rebuttal, and then selecting 'Show negation' from the popup menu. The enthymeme is then displayed as an implicit Warrant. It is then possible to create arguments that support that implicit Warrant as usual. To hide the implicit Warrant you can select 'Hide negation' from the same pop upmenu. Note that you cannot hide the implicit Warrant if it in turn has Backing or supporting arguments. If you are just interested in building Toulmin diagrams, it is recommended that you do not use this option as it is likely to reduce the overall clarity of the analysis.

7. Working in Other Styles: Wigmore

A Wigmore diagram is used mainly in the analysis of legal arguments and was first described by J. H. Wigmore in his book *The Science of Judicial Proof*. The Wigmore diagrams in Araucaria are based on the third edition of this book, published in 1937 by Little, Brown & Company.

To move to the Wigmore diagramming style, do one of the following:

- Select the 'Wigmore' tab above the main diagram panel;
- Hit Ctrl and the comma key (',') to cycle leftwards through the style tabs, or Ctrl and the period key ('.') to cycle rightwards;
- On the menu, select 'View – Style >' and choose 'Wigmore'.

Changing the style of diagram does not affect the stored information about the analysis (technically, all styles are represented and saved using AML). So you can make a change in one style, then swap to another style and make another change, and so on. Most changes will be visible in other styles, and all changes are stored once made.

There are some differences between what can be achieved using the Standard style and what can be achieved using the Wigmore style. Some things can only be done using the Wigmore style. These include:

- Setting premise roles, including testimonial, circumstantial, corroborative and explanatory roles
- Setting fact types, including judicial and tribunal types

These do not have a place in Standard or Toulmin diagrams. As a result, you will see that some toolbar icons and menu options become inactive under the Wigmore style.

Some things are done a little differently in the Wigmore style (evidential types and node labelling, for example). But otherwise, all the basic mechanisms for Standard diagrams also work for Wigmore diagrams.

As with Toulmin diagrams, there are some interesting problems in translating between Wigmore diagrams and the other diagram types. Some of these form the topics of research papers at `http://araucaria.computing.dundee.ac.uk`. This section describes how to use the Wigmore style independently of the Standard and Toulmin styles.

Constructing a Wigmore Diagram

Creating nodes, selecting diagram components, and deleting diagram components works exactly the same as for Standard style diagrams (see sections 2 – 3).

Connecting components is mostly the same. Wigmore diagrams involve nodes that play different **roles**, which work in just the same way as roles in Toulmin diagrams. Wigmore diagrams are constructed from premises of three types:

- Evidential data
- Explanatory data
- Corroborative data

The first of these, evidential data, is subdivided into two forms:

- Testimonial evidence
- Circumstantial evidence

Both of these two forms can be positive or negative, or as Wigmore refers to them,

- Affirmatory evidence
- Negatory evidence

Each of the six types (Testimonial affirmatory/negatory evidence, Circumstantial affirmatory/negatory evidence, Explanatory data and Corroborative data), can be submitted by Prosecution or Defence. This yields twelve Wigmore roles in total:

- Testimonial affirmatory evidence for the prosecution
- Testimonial negatory evidence for the prosecution
- Circumstantial affirmatory evidence for the prosecution
- Circumstantial negatory evidence for the prosecution
- Explanatory data for the prosecution
- Corroborative data for the prosecution
- Testimonial affirmatory evidence for the defence
- Testimonial negatory evidence for the defence
- Circumstantial affirmatory evidence for the defence
- Circumstantial negatory evidence for the defence
- Explanatory data for the defence
- Corroborative data for the defence

The different roles appear as different diagrammatic components: Testimonial evidence is indicated by a box; Circumstantial evidence is is indicated by a circle; Explanatory data is indicated by a right-pointing wedge, and Corroborative data is indicated by a left-pointing wedge. Negatory evidence is marked by leaving open the bottom of the square or circle. Defence nodes are indicated by a double line at the top.

ID Labels

In Standard and Toulmin styles, nodes are usually labelled A, B, C, ..., AA, AB and so on. The convention in Wigmore diagrams, however, is to mark nodes numerically: 1, 2, 3, etc. Whenever Wigmore style is selected, new nodes are added sequentially from the highest free number. (Notice that this means that if you swap from some other style, you may have nodes labelled with a mixture of letters and numbers).

As in Standard, it is possible in Wigmore style to modify node IDs (see Section 4), but the rules that fix what constitutes a valid ID are a little different. You can change the ID to any integer (i.e. any whole number) or any real (i.e. any number accurate to any number of decimal places). You cannot mix alphabetical with numerical numbering in a single ID, but a single diagram can have examples of both.

Examples of valid node IDs:

- A
- Ab
- 5
- 9999
- 0.6
- 99.1

Examples of invalid node IDs:

- ABC
- 99a
- 99.1.2

Changing Roles

Whenever a new node is added to the diagram, it is given the role Testimonial affirmatory evidence for the prosecution. You can change this by doing the following:

- Right-click on the node in the diagram you want to change, and select 'Wigmore role'
- Select the role you want to assign to the node

You can change the role of any node at any time.

Notice that all nodes of a given type (evidential, explanatory and corroborative) are automatically joined together. This is a feature of Wigmore diagrams. Notice too that not all roles are available to all nodes. For example, an explanatory node cannot have corroborative support, and vice versa.

The symbol for the current role of a node is shown in the upper left corner of the box in Full Text view, and is used as the main node symbol in the other views.

If you make a mistake, you can undo, delete and redo work, as in Standard style.

Facts

Wigmore diagrams pay special attention to premises in all roles that are **facts** – that is, that have no further supporting evidence. Any node that is at the very end of a line of reasoning (i.e. any node that has no supporting testimonial or circumstantial evidence, and no explanatory or corroborative data) is a fact.

It is possible to indicate how a fact has been introduced to a case, whether by the judge (a **judicial** fact) or as evidence presented to the court (a **tribunal** fact).

You can set the type of a fact by doing the following:

- Right-click on the node in the diagram you want to change, and select 'Wigmore fact'
- Select the fact type (judicial or tribunal) that you want to assign to the node

Remember that only facts – at the very edges of an analysis diagram – count as facts and can be changed in this way. If you drag further support onto a fact, it ceases to be a fact and will therefore lose its fact type.

In all diagram views, tribunal facts are marked by a pilcrow and judicial facts by a lemniscate.

Belief

Wigmore diagrams allow analysts to mark the degree of **belief** held in a given piece of information or evidence. This is similar to an evaluation in a Standard analysis (see Section 4), but is restricted to just a small number of values:

- Strong belief
- Belief
- Doubt
- Disbelief
- Strong disbelief

You can change the degree of belief in a given node by doing the following:

- Right-click on the node in the diagram you want to change, and select 'Wigmore belief'
- Select the degree of belief you want to assign to the node and click 'Add Evaluation'

Alternatively, you can accomplish the same effect by selecting the node and then doing one of the following:

- Hit 'Ctrl-K'
- One the menu, select 'Labels – Modify evaluation'

You can remove degree of belief from a node entirely by doing any of the following:

- Right-click on the node in the diagram you want to change select 'Wigmore belief' and click 'Delete Evaluation'
- Select the node, hit 'Ctrl-K' and click 'Delete Evaluation'
- Select the node, and then on the menu select 'Labels – Modify evaluation' and click 'Delete Evaluation'

Force

The support arrow or line leading out of a node can be labelled with the **force** it lends to the argument. The force is an indication of the strength of support provided by the statement, and also, for evidence nodes, whether it is **affirmatory** (supports the conclusion) or **negatory** (argues against the conclusion). Force is similar to an evaluation in a Standard analysis (see Section 4), but is restricted to just a small number of values:

- Provisional affirmatory
- Strong affirmatory
- Conclusive affirmatory
- Provisional negatory
- Strong negatory
- Conclusive negatory

The force can be set by doing the following:

- Right-click on the line in the diagram you want and select 'Wigmore force'
- Select the force you want to assign to the line
- Check the “negatory” box if you want the force to be negatory
- Click 'Add Evaluation'

Alternatively, you can accomplish the same effect by selecting the line and then doing one of the following:

- Hit 'Ctrl-K'
- One the menu, select 'Labels – Modify evaluation'

You can remove force from a line entirely by doing any of the following:

- Right-click on the line in the diagram you want to change, select 'Wigmore force' and click 'Delete Evaluation'
- Select the line, hit 'Ctrl-K' and click 'Delete Evaluation'
- Select the line, and then on the menu select 'Labels – Modify evaluation' and click 'Delete Evaluation'

In addition to defining forces on supports arising out of single nodes, the aggregate force of a group of nodes can also be set. If there are two or more supporting nodes of the same type (e.g. two or more evidence nodes), then the net force of this group of nodes can be set by right-clicking on the single edge connecting this group of nodes to the parent node. In addition to the forces available to individual nodes, such aggregate nodes can also be marked with **net probative value**, which covers the following five additional force evaluation labels:

- Strong Inference affirmatory
- Weak Inference affirmatory
- No Inference affirmatory
- No Inference negatory
- Weak Inference negatory
- Strong Inference negatory

Aggregate force values, and net probative value, is assigned in exactly the same way as the other force values. But remember that they can only be applied to aggregation lines of support.

Technical note: When a support edge is given negatory force, Araucaria interprets this as a refutation in Standard mode or a rebuttal in Toulmin mode. To provide consistency with Standard and Toulmin diagrams, an implicit 'added negation' node is added to the Standard diagram which is interpreted as an implicit specialized warrant in the Toulmin diagram (see the discussion in the technical note in the section on Toulmin diagrams). As in the Toulmin case, these added negations are not normally visible in a Wigmore diagram and are hidden by default. The implicit node can be viewed in Wigmore by right-clicking on the node giving the negatory force and selecting 'show negation'. The implicit node can be hidden again by right-clicking on the original node and selecting 'hide negation'. If you are interested in creating Wigmore diagrams, you can ignore this feature and leave all added negations as hidden. Showing them in the diagram is likely to reduce the overall clarity of the analysis.

8. Argument Properties

Araucaria can provide a brief summary of the properties of an argument and its analysis. To access this information for the argument you are analysing,

- On the menu, select ‘File - Properties’

This will open the properties window.

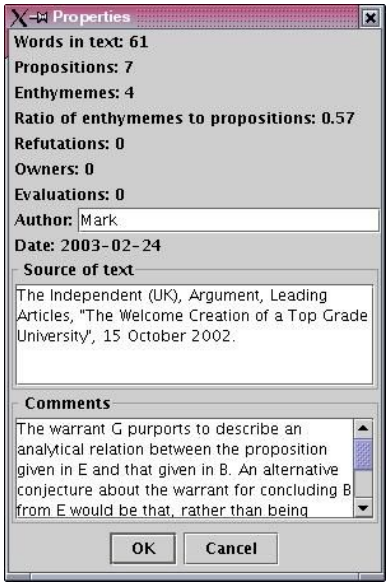

Figure 6. The Properties Window

The top half provides some simple summary information, including:

- The number of words in the original text of the argument
- The number of propositions that have been identified in the original text
- The number of propositions that have been added to the original as enthymemes during reconstruction and analysis
- The ratio of such enthymemes to explicit propositions (this gives a rough indication of the extent of reconstruction that has been carried out by the analyst)
- The number of propositions that have been identified as refutations
- The number of owners that have been used in the argument analysis
- The number of evaluation tags that have been used in the argument analysis
- The date on which the argument analysis was last saved

In addition, there are several fields that can be changed.

The first is the **author** field. By default, this is set to "null", but you can change it to record you own name as the author of the analysis. This data is saved in the AML.

The next two fields allow you to add extra information that is stored in the database and in the AML.

The **source** field allows you to record the source of the original text of the argument. Many of the arguments in the online database also have an online source, so that you can refer back to the original.

The **comments** field allows you to add a description of interesting, controversial, complex or ambiguous featuers in the argument or its analysis. Many of the arguments in the online database have a brief commentary describing analytical choices.

9. Connecting to the AraucariaDB Online Repository

One exciting use of Araucaria is in building an online repository of AML arguments that anyone can access for pedagogic or research purposes. This section describes how you can access this repository, searching for particular forms of argument, and how you can contribute your own analyses to the repository for others to use.

Remember that AraucariaDB is running on a machine at the University of Dundee, so as with any long distance internet connection, you may have to wait a little while if you request substantial amounts of data.

Registering with AraucariaDB

In order to use the AraucariaDB repository, we ask that you register with us. To access the registration window, use the menu and select ‘AraucariaDB - Register’.

First, pick a username: this must be unique, and Araucaria will ask you to try again if you select a name which is already in use. Then fill in your full name, your institutional address and your email address.

You only need to register with AraucariaDB once. Note that no password is required in the current version of Araucaria.

The information with which you supply us is stored on a secure server and will not be passed on to any third party.

Logging on to AraucariaDB

Once you have registered, you will be able to log on to AraucariaDB whenever you run Araucaria. To log on, select ‘AraucariaDB - Log on’ on the menu. You will then be prompted for your username.

Once you have logged on you will be able to search AraucariaDB for arguments, and save your own analyses.

(Note that your username will become the default value for the author field, viewable in the Properties window).

Searching AraucariaDB

Make sure you are logged on. Then, to open up the search window, do one of the following:

- Hit 'Ctrl-Q'
- Click the  icon on the toolbar
- On the menu, select 'AraucariaDB - Search database'

The search window provides three different ways of searching AraucariaDB, the online repository of arguments:

- by text matching
- by structure matching
- by scheme matching

Text matching

The first tab in the search window allows you to enter a text string. When you click on 'Search', Araucaria will connect to AraucariaDB and will return all arguments that include the text you have specified. For example, if you enter "*abortion*", all those arguments that mention abortion will be returned.

Notice that the text matching is simple - Araucaria does not currently support regular expression matching.

Structure matching

The second tab in the search window allows you to enter an **argument fragment**. Clicking on the  icon then asks Araucaria to return all those arguments in the repository which include that fragment. So, for example, you could draw an argument diagram in which a conclusion is supported by one pair of linked premises, and one pair of convergent premises (perhaps to demonstrate to a class the difference between the two forms). AraucariaDB would then return all those arguments in which that structure appears.

To draw the argument fragment that is used as the basis of the search, a miniature version of the Araucaria diagram window is used.

- To add a node, simply click on the pale yellow area
- To delete a node, select it and click the icon
- To link two premises, select them and click the icon
- To unlink two premises, select them and click the icon
- To convert a premise to a refutation (or back) click the icon
- To clear the argument fragment diagram and start again, click the icon

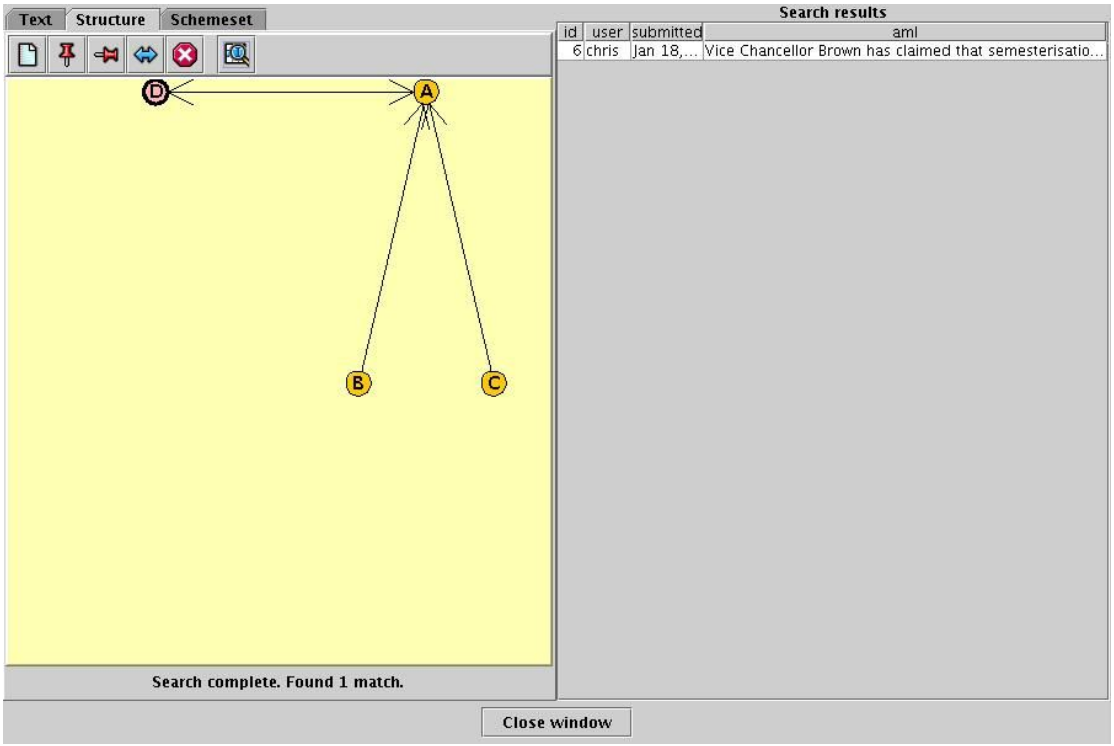

Figure 7. The DB Search Window showing Structure matching

Scheme matching

The third tab in the search window allows you to search the repository for examples of particular schemes. You can select a scheme from the pull down list, or alternatively, if the scheme you are interested in is not in the current schemeset (see section 5 for details), you can type in the name of the scheme manually. When you click ‘Search’, Araucaria will query the repository for all arguments which include the scheme you have specified.

To cancel your search request, click ‘Close Window’. Note that due to software limitations, Araucaria cannot currently interrupt a search once it has commenced.

Whichever method you used to search the database, the matching arguments will be returned to Araucaria, and a summary list displayed in the right hand panel.

To view the arguments provided by AraucariaDB, close the search window, by clicking ‘Close Window’. You will then notice that all the arguments returned by your search are listed in the pull down list on the toolbar. For each argument, the first part of the conclusion is given. To view a particular argument, simply select it from the pull down list. Once selected, the argument is loaded from the database.

You can modify arguments provided by AraucariaDB, but this does not alter the version stored in the database. You can store your modified analysis back to AraucariaDB if you want. These will then be saved as separate arguments.

AraucariaDB can thus store multiple analyses of a given argument.

Submitting Analyses to AraucariaDB

Your own analyses, both those developed in your research and those developed for teaching purposes, may be of use to others. In this way, the duplication of effort can be reduced, and a substantial online resource can be built up.

Remember that to save to AraucariaDB, you need to be logged on (see above).

Once logged on you can save an analysis by doing one of the following:

- Hit ’Ctrl-B’
- Click the  icon on the toolbar
- On the menu, select ‘AraucariaDB - Save to DB’

Your argument analysis is then saved to AraucariaDB, and is logged with your username and the date.

Users cannot currently delete entries from the database. If you wish to delete analyses, please email the project at `araucaria@computing.dundee.ac.uk`

Note that it is also possible to access the AraucariaDB online database using aweb browser. Access is currently limited to searching by text, scheme use, date, user and source. For more details, see

`araucaria.computing.dundee.ac.uk`

10. Using Araucaria in Teaching

One of the uses of Araucaria is in teaching critical thinking, argumentation theory and informal logic. Starting with version 3.0, tools aimed specifically at teaching are being incorporated to make life easier for teachers and students alike.

There is a rudimentary system for automatically marking student analyses by comparing them with model answers. To engage this functionality, from the main menu select 'File – Preferences', click the 'Tutoring' tab, and make sure the 'Tutor mode' box is checked. After clicking OK you will notice a new item on the main menu: 'Tutoring'.

There are two steps to the process, defining the model answer, and comparing against the model answer.

Defining the Model Answer

The tutor builds an analysis as usual. For any given proposition in the analysis, the stretch of text shown in the node (i.e. the stretch of text highlighted, dragged and dropped) indicates the minimum amount possible for a student to have correctly identified this premise. So, for example, if the model answer identifies a premise of the text *should be avoided at all costs*, then a student must also analyse their premise from the first *s* of *should* to the last *s* of *costs*.

Often, however, a tutor will want to mark in such a way that if a student misses a character or two, or even a word or two, they will still be marked as having correctly identified the premise. To do this, select 'Tutoring – Premise endpoints' from the main menu.

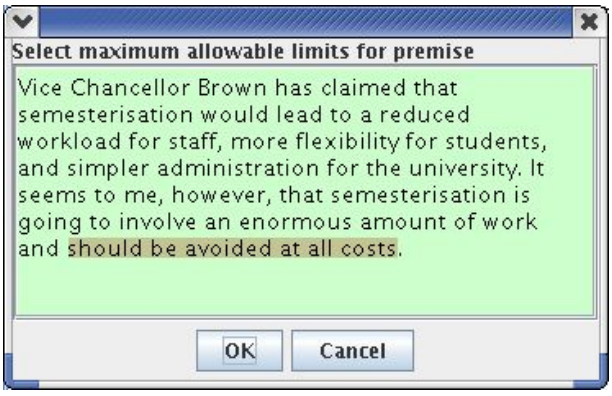

Figure 8. Premise endpoint marking

The tutor's initial selection is shown highlighted. In addition, you can now drag from the earliest possible start point of the proposition (say, in Figure 8, from the*a* of *and*) to the last possible end point (say, the full stop after*costs*).

Between the initial analysis (which represents the minimum), and the highlighted endpoints (which represent the maximum), there is now a range of acceptable analyses for a given premise.

You can save this analysis as a conventional AML file, which will also include information about the endpoint ranges.

Comparing Against the Model Answer

To automatically mark one AML file (a student's submission) against another (the tutor's model answer), select 'Tutoring – Marking' from the main menu.

Regardless of the file currently loaded into Araucaria, you must first load the model answer by clicking the 'Tutor' button, and then load a given student file by clicking the 'Student' button. Finally, click on the 'Mark' button.

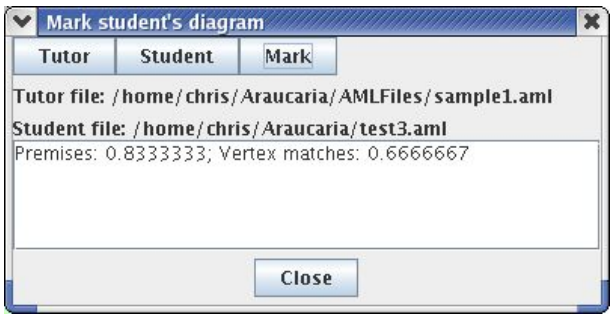

Figure 9. Marking

The results are shown in terms of the number of premises that were identified in the same way in the two files (within the limits defined by the tutor), and the number of structural similarities (so, for example, the same two premises supporting the same conclusion would be a match). Both figures are shown as a percentage.

The same technique can be used to compare any two arbitrary AML files.

It is possible to use Araucaria to automatically process an email box full of student submissions and collate marks into a spreadsheet. For more information, contact

`araucaria@computing.dundee.ac.uk`

11. System Settings

There are a number of features in Araucaria that can be configured by the user. To access these settings,

- On the menu, select ‘File - Preferences’

This opens the Preferences window.

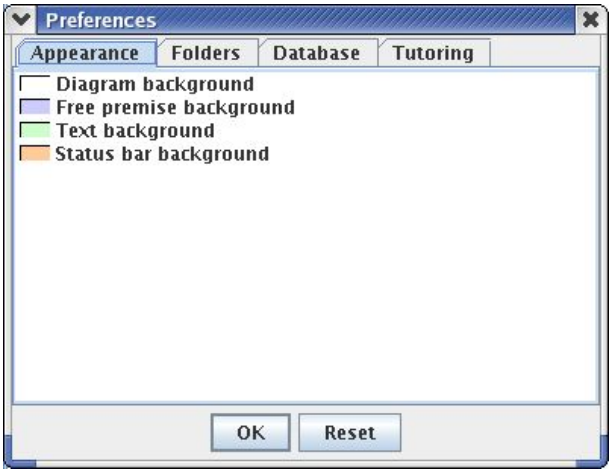

Figure 10. The Preferences Window

There are four sections or “tabs”.

The first, Colours, allows you to change the colours of the main areas of the Araucaria window to suit your taste.

The second, Folders, allows you to configure where Araucaria looks for your files.

The third, Database, allows you to change the default settings for the AraucariaDB database.

The fourth, Tutoring, allows you to enable the tutoring-specific features of Araucaria.

Changing Colours

To change the colour of any of the four main panes in Araucaria (The diagram area on the right, the small area for newly dropped nodes just below it, the text area on the left, and the message area to the bottom), open the Preferences window, click the "Colours" tab, and then click once on one of the four pane names.

You can then change the colours associated with the background and the text that appears in that pane. Click 'OK' to confirm the changes, or 'Cancel' to abort. Hitting 'Reset' will return the settings to their original colours (black text in a pale blue window for text, a pale yellow window for newly dropped nodes, a white window for diagrams and a yellow window for messages).

Changing Folders

Araucaria loads and saves various files including text files for analysis, AML files for analysed arguments and schemesets of argumentation schemes.

Although when you open and save such files, you can always select wherever you want the files to be, it can save time if you have directories where you always keep such files.

By default, Araucaria always looks in the directory into which it was installed, in sub-directories called TextFiles, AMLFiles and SchemeFiles. You can change these default directories for each of the three types of file, by opening the Preferences Window, selecting the 'Folders' tab, and changing the values in the textboxes.

Click 'OK' to confirm, 'Cancel' to abort, or 'Reset' to set all preference values back to the original defaults.

Changing AraucariaDB Access Properties

By default, Araucaria can connect to the AraucariaDB online database running at the University of Dundee. *You should not normally need to change these settings.*

It is possible to change either the way in which Araucaria connects to Dundee, or even to set up your own local database and configure Araucaria to communicate with that instead of talking to Dundee. This may be useful if you want your students to build up a set of examples at your institution, for example.

If you wish to do this, we can help you set up a database with the correct structure, though you will probably need to involve the systems administrator at your site to help you configure an appropriate server machine. For further information, please contact

`araucaria@computing.dundee.ac.uk`

Enabling Tutoring

To enable tutoring mode, select ‘File – Preferences’ and click the ‘Tutoring’ tab. Click to tick the ‘Tutor mode’ check box.

See section 10 for more information.

All of the Preferences you confirm by clicking ‘OK’ will be stored for the next time you run Araucaria.

12. Argument Markup Language

The Argument Markup Language (AML) has been designed using XML to offer a flexible, open, portable standard for describing arguments.

XML languages are described in a Document Type Definition (DTD), against which files such as arguments in AML can be verified. Araucaria both ensures that valid AML files are output, and verifies any AML files loaded. Araucaria will read any valid AML files, regardless of whether or not they were generated by Araucaria.

The AML provided with Araucaria v3.0 and v3.1 is an improvement upon, and an extension of, that provided with earlier versions. Because Araucaria enforces strict validity rules, Araucaria v3.0 may not be able to read AML files created by Araucaria v1.0. There are several ways to circumvent this problem; please contact the research team for further information. There should be no compatibility problems between Araucaria v2.0 and Araucaria v3.0, nor between v3.0 and v3.1.

Note that a `.aml` extension simply indicates a text file containing XML marked up text conforming to the document type definition `argument.dtd`. Schemeset files have a `.scm` extension, and store a single `<SCHEMESET></SCHEMESET>` unit, as defined in `argument.dtd`.

For details of other software that exploits AML, please see `www.computing.dundee.ac.uk/staff/creed/`

The AML DTD is included here for reference. Any queries should be directed to `araucaria@computing.dundee.ac.uk`

```
<!-- ************************************
     *                                  *
     *   argument.dtd                   *
     *                                  *
     *   XML DTD for Argument Markup    *
     *   Version 1.5.1                  *
     *                                  *
     *   17 June 2004                   *
     *                                  *
     *   Copyright 2001 - 2005          *
     *   Chris Reed & Glenn Rowe        *
     *   Department of Applied Computing  *
     *   University of Dundee           *
     *                                  *
     *   Released under the             *
     *   GNU General Public License     *
     *                                  *
     ************************************
-->

<!-- ARG
     Topmost element
     TEXT and SCHEMESET children are both optional
     The AU element, which corresponds to the root node,
     is also optional to allow schemeset files to be valid
     AML.
-->
<!ELEMENT ARG (SCHEMESET?, TEXT?, EDATA?, AU?)>

<!-- TEXT
     The original text of the argument
-->
<!ELEMENT TEXT (#PCDATA)>

<!-- AU
     An argument unit, composed of a proposition (a conclusion)
     followed, optionally, by a single refutation, followed,
     optionally, by premises arranged in either convergent or
     linked structures in any order.
-->
<!ELEMENT AU (PROP, REFUTATION?, (CA | LA)*)>

<!-- PROP
     A proposition (premise or conclusion) within the argument.
     The text is available in PROPTEXT; the INSCHEMES are
     the schemes of which this node is a member.
     The TUTOR element is present if this PROP is part of a tutorial
     question.
     Attributes:
        - identifier is optional; browsers should be prepared to
          generate identifiers if they are not available in the XML.
          If identifiers are used, they must be consistent (i.e.
          each proposition must have unique identifier) hence ID type.
        - missing: used to indicate premises which are left implicit.
        - nodelabel. A label, just as for support label, but
          attached to the proposition node, rather than an edge.
          Valid AML can contain either nodelabels, or supportlabels,
          or a mixture of both, or neither. Individual applications
          may wish to calculate one (e.g. proposition values) from
          the other (e.g. support values), and propagate values
          through the graph.
        - supportlabel. A label attached to the edge leading from the
          node to its parent. This attribute is optional, and
          will be ignored for the root node. Typically this
          label takes a value drawn from a small dictionary
          to express an evaluative position with respect to the
          support contributed by this argument. The label can
          also be probabilities, allowing the construction of,
          e.g., Bayesian nets, from the AML.
-->
<!ELEMENT PROP (PROPTEXT, OWNER*, INSCHEME*, ROLE*, TUTOR?)>
<!ATTLIST PROP identifier   ID         #IMPLIED
               missing      (yes | no) "no"
               nodelabel    CDATA      #IMPLIED
               supportlabel CDATA      #IMPLIED>

<!-- ROLE
    An optional label used to identify roles in non-Standard diagrams, such
    as Toulmin or Wigmore.
        - class: The diagram type (e.g. Toulmin or Wigmore)
        - element: The node type (e.g. data, warrant, etc in Toulmin)
-->
<!ELEMENT ROLE EMPTY>
<!ATTLIST ROLE class        CDATA      #REQUIRED
               element      CDATA      #REQUIRED>

<!-- PROPTEXT
     The text associated with a given node or proposition.
     Attributes:
```

```
        - offset. Number of characters into text specified in TEXT
          section to which this proposition corresponds. Optional:
          if omitted, browsers should open the argument read-only.
          If no TEXT section is specified, this offset should be ignored.
          If the PROP is MISSING=YES then offset should be ignored.
-->
<!ELEMENT PROPTEXT (#PCDATA)>
<!ATTLIST PROPTEXT offset CDATA #IMPLIED>

<!-- OWNER
     The name of a party who is attributed with the proposition.
     One proposition may be owned by many parties - or may not
     be registered as being owned by any party.
     The details of the party are specified in the attribute -
     notice that OWNER is an empty element and does not therefore
     need any further data.
     Attributes:
        - name. The name of the party (names are assumed to be
          unique, but one party may 'own' many propositions).
-->
<!ELEMENT OWNER EMPTY>
<!ATTLIST OWNER name CDATA #REQUIRED>

<!-- INSCHEME
     A scheme of which the current proposition is a member. The
     details of the scheme are specified in two attributes, both
     of which are required. For multiple schemes, use one
     INSCHEME element for each scheme. Note that INSCHEME is an
     empty element and does not therefore need any further data.
     Attributes:
        - scheme. Textual name of an argument scheme which should
          match a scheme in the scheme set used by the browser or
          specified in the SCHEMESET section
        - schid. Identifier to a particular scheme in this arg:
          browsers are expected to be able to handle more than one
          occurrence of a given scheme at a given node.
-->
<!ELEMENT INSCHEME EMPTY>
<!ATTLIST INSCHEME scheme CDATA #REQUIRED
                   schid  CDATA #REQUIRED>

<!-- TUTOR
     If a PROP is part of a tutorial question in which the student
     is required to mark up some text and build an argument tree, we
     must allow for variable start and end points for the text within
     a PROP. The TUTOR element contains attributes which allow these
     endpoints to be defined.
     Attributes:
        - start. The offset into text specified by TEXT which is the
          earliest point in the text at which the student is allowed to
          start the premise. The latest start point is provided by the
          offset attribute in PROPTEXT. It is assumed that any point
          between TUTOR.start and PROPTEXT.offset is acceptable as a
          starting point for the premise, even though some of these points
          could be in the middle of a word. If 'start' is missing, it should
          be taken to be the same as PROPTEXT.offset. (That is, there is
          only one place that is acceptable as a start point.)
        - end. The offset into TEXT that is last acceptable endpoint of
          the premise. The earliest acceptable endpoint of the premise
          is provided by PROPTEXT.offset + the length of the PCDATA component
          of PROPTEXT. As with the start point, any point between the earliest
          and latest acceptable endpoints is allowed as the endpoint of a
          premise. If 'end' is missing, it should be taken to be the same as
          PROPTEXT.offset + length(PROPTEXT).
-->
<!ELEMENT TUTOR EMPTY>
<!ATTLIST TUTOR start CDATA #IMPLIED>
<!ATTLIST TUTOR end CDATA #IMPLIED>

<!-- REFUTATION
     A proposition which expresses the converse of a proposition.
     In a formal system, this would simply be the negation, but
     in real language, a more flexible, individually specified
     proposition is more appropriate. Notice that links between
     propositions are always links of support. To characterise
     rebutting, undercutting and refuting links, it is necessary
     to introduce a refutation proposition. Thus if A refutes B,
     introduce C (or not B, if you like) as the refutation of B,
     then support C with A.
-->
<!ELEMENT REFUTATION (AU)>

<!-- CA
     A convergent argument. The PROP specified immediately previously
     is its conclusion, and the PROP within its body is its premise.
     A convergent arg has exactly one premise (other convergent args
     may have the same conclusion, of course).
-->
<!ELEMENT CA (AU*)>
```

```
<!-- LA
     A linked argument. The PROP specified immediately previously
     is its conclusion, and the PROPs in its body are its premises.
     A linked argument must have two or more premises.
-->
<!ELEMENT LA (AU, AU+)>

<!-- SCHEMESET
     The first section of an ARG, coming before the TEXT and the
     structure supporting the topmost conclusion. Includes any number
     of SCHEME definitions. There is no requirement that any or
     all of these schemes be used in the argument that follows.
     The Araucaria software assumes that any schemes used in the
     argument are defined in this section.
-->
<!ELEMENT SCHEMESET (SCHEME)*>

<!-- SCHEME
     The definition of an argument scheme, with a name (potentially
     used in the scheme= attribute of a PROP) and any number of
     critical questions.
-->
<!ELEMENT SCHEME (NAME, FORM, CQ*)>

<!-- NAME
     The name of a scheme. These must be unique (though there may be
     multiple occurrences of any one scheme in an argument).
-->
<!ELEMENT NAME (#PCDATA)>

<!-- FORM
     The description of a scheme - its premises (any number) and
     conclusion (exactly one)
-->
<!ELEMENT FORM (PREMISE*, CONCLUSION)>

<!-- PREMISE
     A premise in an argumentation scheme
-->
<!ELEMENT PREMISE (#PCDATA)>

<!-- CONCLUSION
     The conclusion of an argumentation scheme
-->
<!ELEMENT CONCLUSION (#PCDATA)>

<!-- CQ
     A critical question.
-->
<!ELEMENT CQ (#PCDATA)>

<!-- EDATA
     Extended data, including material external to the argument.
     All extended data is optional.
-->
<!ELEMENT EDATA (AUTHOR?, DATE?, SOURCE?, COMMENTS?)>

<!-- AUTHOR
     The author of this analysis (not of the original argument)
     (Remember that a given source argument may have multiple
     analyses)
-->
<!ELEMENT AUTHOR (#PCDATA)>

<!-- DATE
     The date the analysis was carried out (not the date of the
     original argument)
     Though not enforced by the DTD, this is assumed to be in
     YYYY-MM-DD format.
-->
<!ELEMENT DATE (#PCDATA)>

<!-- SOURCE
     The original source of the argument. Can be a reference
     (e.g. in APA style) or, if possible a URI to the material
     online.
-->
<!ELEMENT SOURCE (#PCDATA)>

<!-- COMMENTS
     Offers space for a commentary on the argument, highlighting
     non-obvious or controversial features of the analysis.
-->
<!ELEMENT COMMENTS (#PCDATA)>
```

13. Menu Reference

File

Open Text File

Loads the text of a new argument to analyse

Open Argument

Loads a previously analysed (or partially analysed) argument stored in AML format.

Save Argument

Save an analysed (or partially analysed) argument in AML format

Save Argument As

Save an analysed (or partially analysed) argument in AML format under a new filename

Save Diagram

Store the diagram representing the analysis in JPG or TIF format

Close All

Close the current argument analysis

Properties

Access summary statistics and the source and comments fields for the current argument analysis

Preferences

Access and modify the current system settings

Exit

Close the Araucaria program

Edit

Undo

Undo the previous operation

Redo

Repeat a previously undone operation

Clear diagram

Restart the analysis, clearing the diagram window and unselecting all text

Flip diagram

Turn the diagram upside down (if the conclusion is at the top, it will be placed at the bottom and vice versa)

Missing premise

Insert a node corresponding to a missing premise (useful in enthymematic arguments

Refutation

Convert the currently selected nodes into refutations of the nodes they support

Delete selected

Remove the selected components from the diagram. If an arrow is selected, then the two nodes at either end are left unconnected. If a node is selected, it is removed and its corresponding text is freed.

Link statements

Join the selected nodes into a linked argumentation structure

Unlink statements

Return the selected linked argumentation structure to a convergent structure

Select all nodes

Select all the **nodes** (not arrows) in the diagram

View

Zoom

Alter how much of the diagram can fit in the window

Scaled – Fit the entire analysis into the window

Full Size – Label nodes with IDs to save space

Full Text – Include the full text in each node for clarity

Style

Show the analysis according to a particular diagramming style

Standard – According to the standard box-and-arrow style

Toulmin – According to the Toulmin style

Pollock – In the style of Pollock's defeasible inference graphs

Wigmore – In the style of Wigmore's jurisprudential charts

Labels

Modify ownership

Change the owners assigned to the selected proposition(s)

Show/Hide owners

Show or hide ownership information in the diagram

Modify evaluation

Change the evaluation assigned to the selected proposition(s)

Show/Hide evaluations

Show or hide evaluation information in the diagram

Schemes

Select scheme

Attach an argumentation scheme to the selected part of the diagram

Add/edit scheme

Modify the current schemeset

Save schemeset

Save the current schemeset in SCM format

Open schemeset

Load a previously saved schemeset in SCM format

AraucariaDB

Login

Connect to the AraucariaDB online repository

Register

Save user details with the AraucariaDB online repository

Save to database

Save the current analysis to the AraucariaDB online repository

Search database

Search the online repository of arguments for examples that match particular patterns

Help

Help

Display a summary help screen

About

Display the splash screen, including contact and license details

14. Toolbar Reference


Open a new text file


Open an argument analysis


Save the argument analysis


Undo last action


Redo last action


Clear the diagram


Invert the diagram


Save the diagram


Insert missing premise


Set/unset selected node as refutation


Delete currently selected items


Link selected nodes


Unlink selected nodes


Select argumentation scheme


Save analysis to AraucariaDB


Search AraucariaDB

Pull-down list box of AraucariaDB search results


Help

15. System Requirements

Araucaria runs in Java, which means that it can run on any machine that can run Java 1.5, including Windows PC's (95, 98, NT4, 2000, me, XP), Linux platforms (all distributions), UNIX machines (running Solaris), Macs (OS X from 10.4 Tiger with the J2SE 5.0 download) and many others besides (see `http://java.sun.com/j2se/1.5.0/system-configurations.html` for details). Further updates and information will be released on the Araucaria homepage,

`http://araucaria.computing.dundee.ac.uk/`

The advantage of using Java is platform independence and open interoperability, which are features to which the Araucaria project is committed. The disadvantage is that on older machines, Araucaria, like other Java programs, may run rather slowly. Because of the demands of Java, we recommend at least a machine with a processor running at 500MHz, with 128Mb RAM, and a monitor capable of SVGA resolution.

Araucaria needs to be installed on your computer. Installation requires around 7 Mb of hard disk space, plus 50-60 Mb for Java 1.5 and resource files. The installation procedure is very straightforward, and is described in section 16.

For connection to the AraucariaDB online repository, you will need an internet connection from your machine. Connection to AraucariaDB has not been tested through complex firewall configurations, so if you experience connection problems, we suggest you contact your network administrator in the first instance (but after that, please feel free to email queries to `araucaria@computing.dundee.ac.uk`).

16. Installation

The distribution CD should start up automatically under Windows, and then guide you through the process of installing to your hard disk. If the CD does not run automatically, use a web browser to access the file `index.html`. In Windows, open 'My Computer' or 'Windows Explorer' and double click the file `index.html` on the CD drive (often D:). The same file can be installed from the CD or from the project webpage:

`http://araucaria.computing.dundee.ac.uk/`

In Windows, Araucaria is supplied as a setup file that will automatically install the Araucaria components in an appropriate place. Simply run the `Araucaria3-VM-setup.exe` program. For other platforms (and indeed, for Windows too), Araucaria is supplied as a self-extracting zip file. The folder that is created contains, amongst other things, the file `Araucaria.sh` (or `Araucaria.exe` on Windows) which you should double-click to run Araucaria. If you experience any difficulties, please email `araucaria@computing.dundee.ac.uk`.

For Mac users, the CD also includes `Java15Release1.dmg`, which you should double click, open the mounted disk, and double click the Installer to install Java 1.5 if you do not already have it (it is NOT included as standard on Mac OS 10.4). Then unzip Araucaria as usual.

To install under other operating systems, unzip from the file `NativeNoVM.zip` and then, run `Araucaria.sh` (note that you may need to tweak this script for your system). You may need to install Java 1.5 manually: for details, see `http://java.sun.com/j2se/faq.html`.

The installation contains the following files and directories:

`/Araucaria.exe`	Laucher (`Araucaria.sh` under *nix)
`/argument.dtd`	AML definition (see section 12)
`/prefs.dat`	Preferences file (see section 11)
`/AMLFiles/`	Sample argument analyses
`/doc/`	Documentation, including this manual in .pdf
`/Help/`	Summary online help
`/images/`	Icons and other images
`/jars/`	Program files
`/jre1.5.0_02/`	Java Virtual Machine (optional)
`/SchemeFiles/`	Schemesets
`/src`	Araucaria source files
`/Textfiles`	Sample arguments

Index