AntMover 0.9 – A Text Structure Analyzer

Overview and User Guide

1.1 Introduction

AntMover 1.0 is a prototype version of a general learning environment that can be applied to the analysis of text structure in any field or discipline, and to any text type. In this user guide, I will explain its main features and an overall guide to its use.

Before giving a detailed account of the system, it should be stressed that as an aid to teachers and learners, *AntMover* can be used in a quite simple manner, and requires little explanation. This is because most of the commonly used functions are intuitively placed on the screen via the system's graphical user interface (GUI), and there is an extensive use of warnings to prevent novice users from inadvertently modifying or deleting essential parts of the system. Ease of use in this context is essential, as it is known that many learners and even teachers may not have well developed computer literacy skills.

On the other hand, *AntMover* is also designed to be a completely general learning system and so knowledgeable users can modify almost all aspects of the system to suit their particular needs. This feature is vital if the system is to be used as a serious aid to research on text analysis. Unfortunately, modifying the system for a particular task is not trivial and so some instruction is required. Saying this, *AntMover* has also been designed to make these changes as apparent as possible, and so it is anticipated that any user with only a short exposure to the system will be able to implement these changes.

To reflect the different ways in which *AntMover* can be used, this guide will first cover some of the essential elements of the system that all users will need to understand. Following this, some of the more complex aspects will be covered. Areas discussed in this guide are as follows:

- *AntMover* 1.0 specifications
- Installing and Uninstalling *AntMover*
- Launching and Exiting *AntMover*
- Basic Operations

- Modifying Decisions made by *AntMover*
- Viewing Training Data and Knowledge Data Sets
- Viewing Project and General Preferences
- Creating a New Project
- Selecting and Modifying an Existing Project

1.2 AntMover 1.0 Specifications

AntMover was developed using the programming language PERL 5.6.1, and a number of modules available through the CPAN (*Comprehensive Perl Archive Network*) at http://www.cpan.org/. These modules are Tk, Win32::FileOp, File::DosGlob, Tk::SplitFrame, Lingua::Eng::Sentence, and Tk::ROText. *AntMover* was developed using the *Komodo 1.2* interactive development environment (IDE) available from ActiveState (http://www.activestate.com/), and compiled into an executable (.exe) file using ActiveState's *Perl Development Kit* (http://www.activestate.com/). Some of the more important specifications of the *AntMover* system are listed below.

OS:	Microsoft Windows 95/98/2000/ME/XP/NT or later.
Hard disk Space Required:	1.8 MB (Self Extracting Compressed File) 1.4 MB for main program files 6.5 MB for main program and all associated files (including demo project)
Memory Requirements:	128 MB (recommended) Note that more memory is required if a large number of files are to be processed at the same time.
Development Language:	Perl 5.6.1 (Build 633 from ActiveState)
Development Environment:	Komodo 1.2.5 (ActiveState)
Distribution:	Self Extracting Compressed File
Language Used:	English

1.3 Installing AntMover

AntMover is distributed as a single self extracting compressed zip file. To install the *AntMover* files and directories, simply double click on the .exe file and expand the zip file onto the user hard-drive After expanding the software, the directory structure shown in Figure 1.1 is created.

Figure 1.1 Directory Structure of *AntMover* Software

```
antmover
        antmover1_0.exe
        antmover_prefs
                 antmover_gen_prefs.prefs
        antmover_projects
                 Information_Technology_Abstracts
                          training_sets
                                   Information_Technology_Abstracts_1.txt
                                   Information_Technology_Abstracts_2.txt
                                   Information_Technology_Abstracts_3.txt
                                   …
                          data_sets
                                   flow_probs_file.txt
                                   label_file.txt
                                   preferences_file.txt
                                   orig_feat.txt
                                   orig_dist.txt
                                   chi_feat.txt
                                   chi_dist.txt
                                   ig_feat.txt
                                   ig_dist.txt
        antmover_training_classes
                 1_claiming_centrality
                          1_12.txt
                          1_132.txt
                          1_163.txt
                          …
```

1.3.1 Important Directories Used by AntMover

antmover:

Holds all files and subdirectories required by the system.

antmover_prefs:

Stores the file holding information about general system preferences.

antmover_projects:

Stores all projects created by the system.

antmover_training_classes:

Stores a set of directories containing raw training data for the system.

Information_Technology_Abstracts:

A predefined project for analyzing abstract structure in computer science.

training_sets:

Stores a set of training files specially formatted for use by the system.

data_sets:

Stores essential files used for processing data in a project.

1_claiming_centrality:

A sample directory that holds a set of training data. Note that the format of the directory name corresponds to a class number followed by an underscore and then the class label with underscores replacing label spacing.

1.3.2 Important Files Used by AntMover

[antmover_1_0.exe]:

The main *AntMover* program.

[antmover_gen_prefs.prefs]

A file storing general system preferences and settings.

[*.*.txt]:

A file used as raw data for training the system. Note that any file name can be used, but it must be formatted as text, i.e., a .txt file.

[orig/chi/ig_feat.txt]:

A file storing a ranked list of features and feature weights used in a project, calculated using either a raw frequency measure (orig), chi-squared (chi), or information gain (ig).

[orig/chi/ig_dist.txt]:

A file storing a 'bag of clusters' representation for each file in the project training data, based on the feature reduction procedure used. Raw frequency measure (orig), chi-squared (chi), or information gain (ig).

[flow_probs_file.txt]:

A file storing probability information about common groupings of step units used in a project.

[label_file.txt]:

A file storing the class names and labels used in a project.

[preferences.txt]:

A file storing the preferences and settings used when creating a project.

[Information_Technology_Abstracts_1.txt]:

An example file used to store a specially formatted version of the first training data file used in the project "Information_Technology_Abstracts".

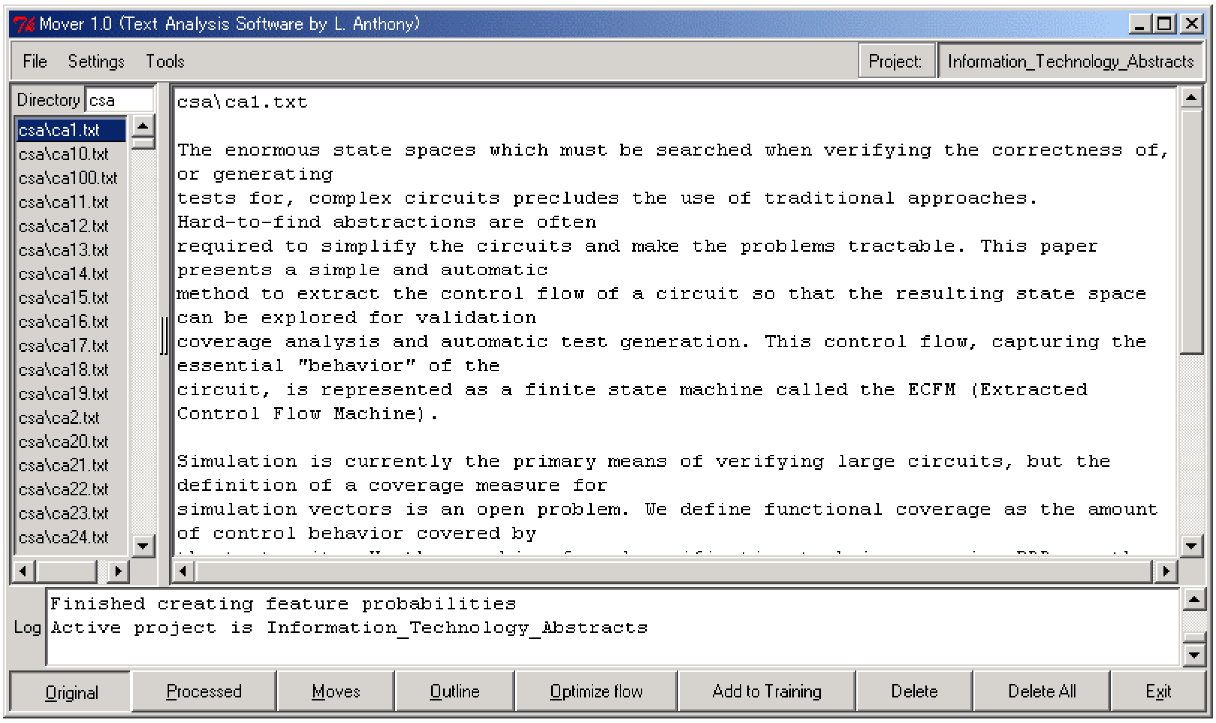
1.4 Uninstalling AntMover

To uninstall *AntMover*, simply delete the directory antmover, and all subdirectories and files.

1.5 Launching AntMover

To launch *AntMover*, double click on the antmover.exe icon in the antmover directory. After launching the program, the 'main window' of *AntMover* will be displayed (Figure 1.2).

Figure 1.2 Main Window of *AntMover*

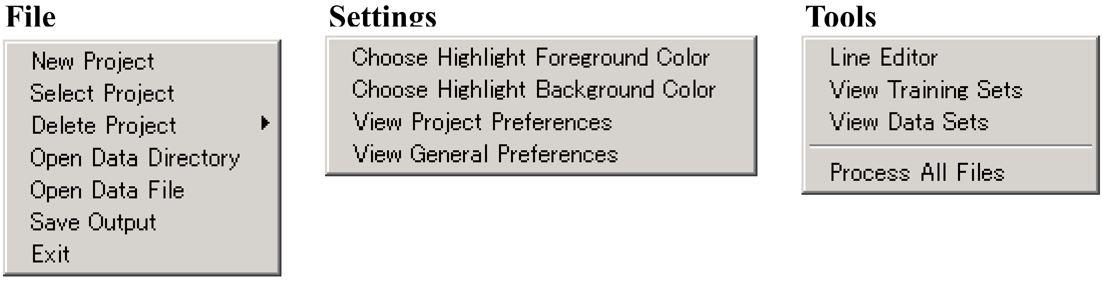


```
csa\ca1.txt

The enormous state spaces which must be searched when verifying the correctness of,
or generating
tests for, complex circuits precludes the use of traditional approaches.
Hard-to-find abstractions are often
required to simplify the circuits and make the problems tractable. This paper
presents a simple and automatic
method to extract the control flow of a circuit so that the resulting state space
can be explored for validation
coverage analysis and automatic test generation. This control flow, capturing the
essential "behavior" of the
circuit, is represented as a finite state machine called the ECFM (Extracted
Control Flow Machine).

Simulation is currently the primary means of verifying large circuits, but the
definition of a coverage measure for
simulation vectors is an open problem. We define functional coverage as the amount
of control behavior covered by
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

The 'Main Window' is composed of five areas. At the top of the window is the 'Menu Bar', which is used for file operations, viewing and changing system settings, and accessing various tools of the system. The options available from the 'Menu Bar' are shown in Figure 1.3.

Figure 1.3 Menus used in *AntMover*



On the left side of the 'Main Window' is the 'File List' frame that shows all current files under investigation. The default directory storing the current files is set in the 'General Preferences Window' (see later). On the right side of the 'Main Window' is the 'Analysis Frame' that displays the results of applying the *AntMover* analysis depending on which viewing option is chosen (see below). Under the 'Analysis Frame' is a small 'Log Frame' that provides a log of the system events as *AntMover* is running. This can be useful for detecting errors and problems encountered during the creation and modification of system projects. The row of buttons or 'Button Bar' at the bottom of the 'Main Window' serve a number of functions, including changing the viewing option of the text under investigation, optimizing the system analysis to account for structural flow, adding texts to the project training data, and deleting files in the 'File List' frame.

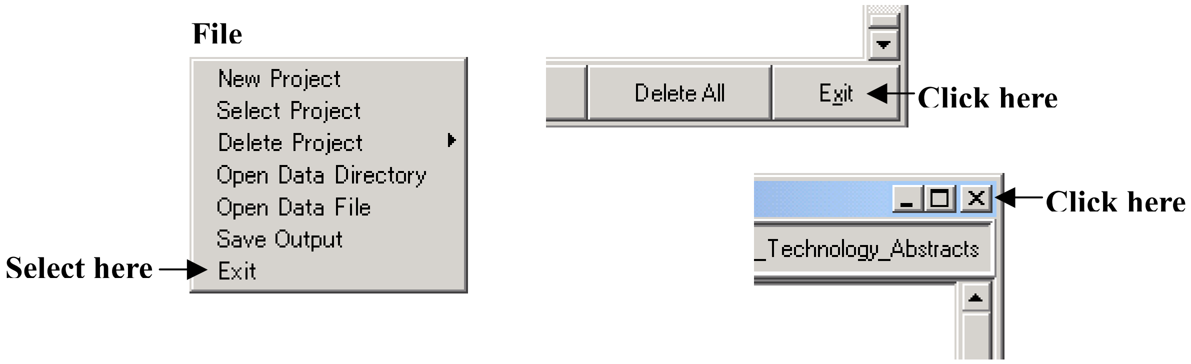
1.6 Exiting AntMover

Exiting *AntMover* can be achieved in one of three ways: 1) Select the "Exit" option from the FILE menu, 2) Click on the Exit button on the right of the 'Button Bar', and 3) Click on the 'close window' icon at the top right of the 'Main Window' (Figure 1.4).

1.7 Getting Started with AntMover

When *AntMover* is launched for the first time, it is setup with the optimum settings to

Figure 1.4 Exiting *AntMover*



conduct an analysis of move structure in RA abstracts of computer science using the 'Modified CARS Model' described in Chapter Four of this thesis. *AntMover* also launches with a set of 600 computer science abstracts with which to try out the system. Using the following procedure, *AntMover* can be quickly used to analyze this set of example texts.

Step 1: Launch AntMover

This will open the 'Main Window' with the list of 600 example texts appearing in the 'List Frame'.

Step 2: Click on a text in the 'List Frame' to analyze the text.

When a text from the 'List Frame' is selected, the original file is displayed in the 'Analysis Frame' in its original state without applying any formatting (Figure 1.5).

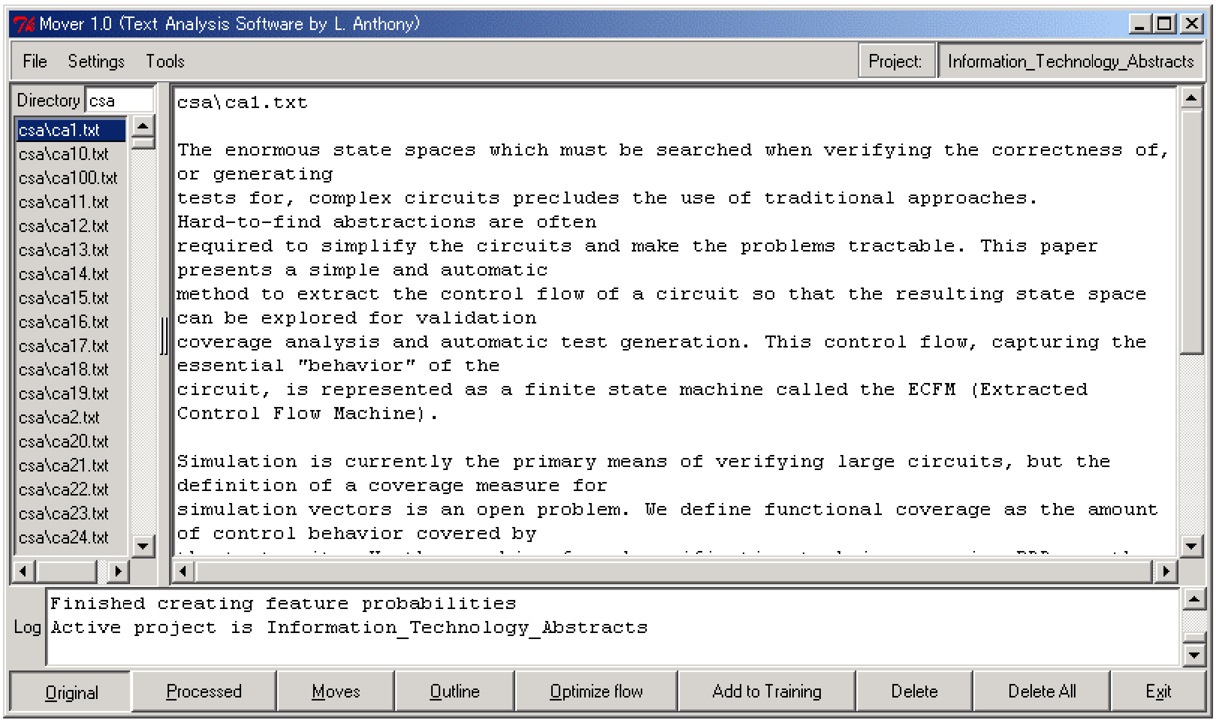
Step 3: Click on the Processed button in the 'Button Bar'

When the *Processed View* is selected, the target text will be automatically processed into step units, and other formatting problems corrected (Figure 1.6).

Step 4: Click on the Moves button in the 'Button Bar'

When the *Moves View* is selected, the step units of the target text will be automatically classified into appropriate steps of the structural model (Modified CARS Model), and

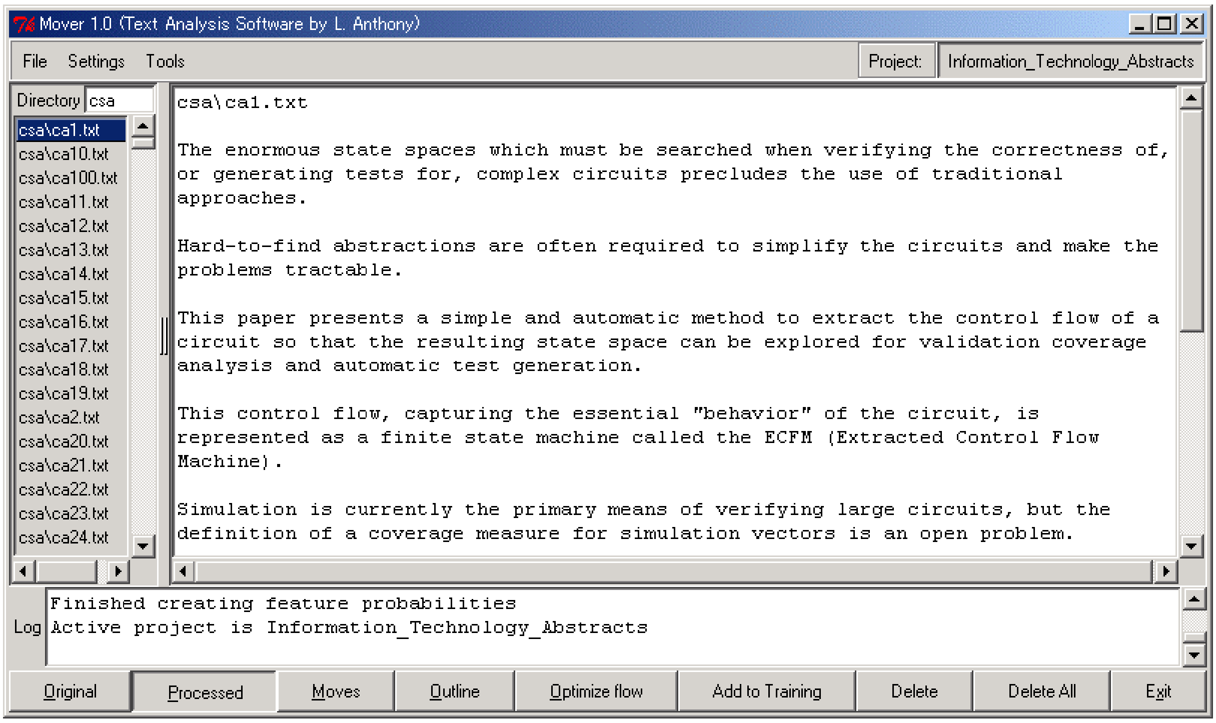
Figure 1.5 *AntMover* in the 'Original' Viewing Mode



```
csa\ca1.txt

The enormous state spaces which must be searched when verifying the correctness of,
or generating
tests for, complex circuits precludes the use of traditional approaches.
Hard-to-find abstractions are often
required to simplify the circuits and make the problems tractable. This paper
presents a simple and automatic
method to extract the control flow of a circuit so that the resulting state space
can be explored for validation
coverage analysis and automatic test generation. This control flow, capturing the
essential "behavior" of the
circuit, is represented as a finite state machine called the ECFM (Extracted
Control Flow Machine).

Simulation is currently the primary means of verifying large circuits, but the
definition of a coverage measure for
simulation vectors is an open problem. We define functional coverage as the amount
of control behavior covered by
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

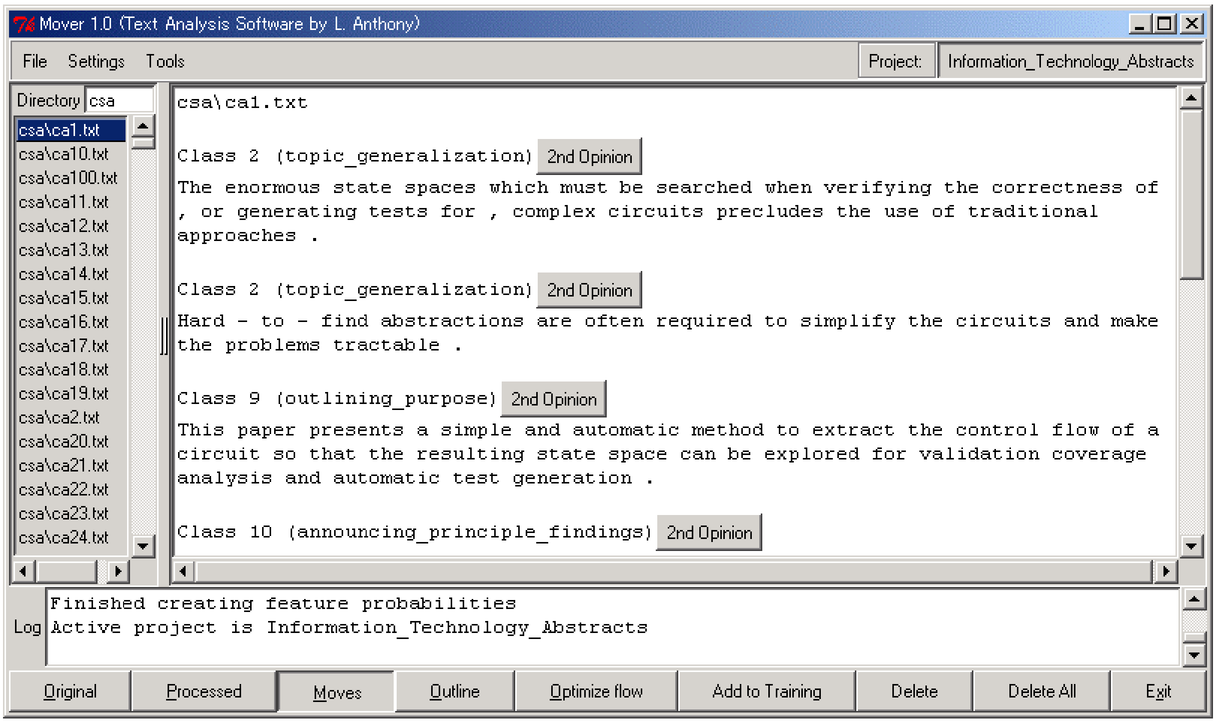
Figure 1.6 *AntMover* in the 'Processed' Viewing Mode



```
csa\ca1.txt

The enormous state spaces which must be searched when verifying the correctness of,
or generating tests for, complex circuits precludes the use of traditional
approaches.

Hard-to-find abstractions are often required to simplify the circuits and make the
problems tractable.

This paper presents a simple and automatic method to extract the control flow of a
circuit so that the resulting state space can be explored for validation coverage
analysis and automatic test generation.

This control flow, capturing the essential "behavior" of the circuit, is
represented as a finite state machine called the ECFM (Extracted Control Flow
Machine).

Simulation is currently the primary means of verifying large circuits, but the
definition of a coverage measure for simulation vectors is an open problem.
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

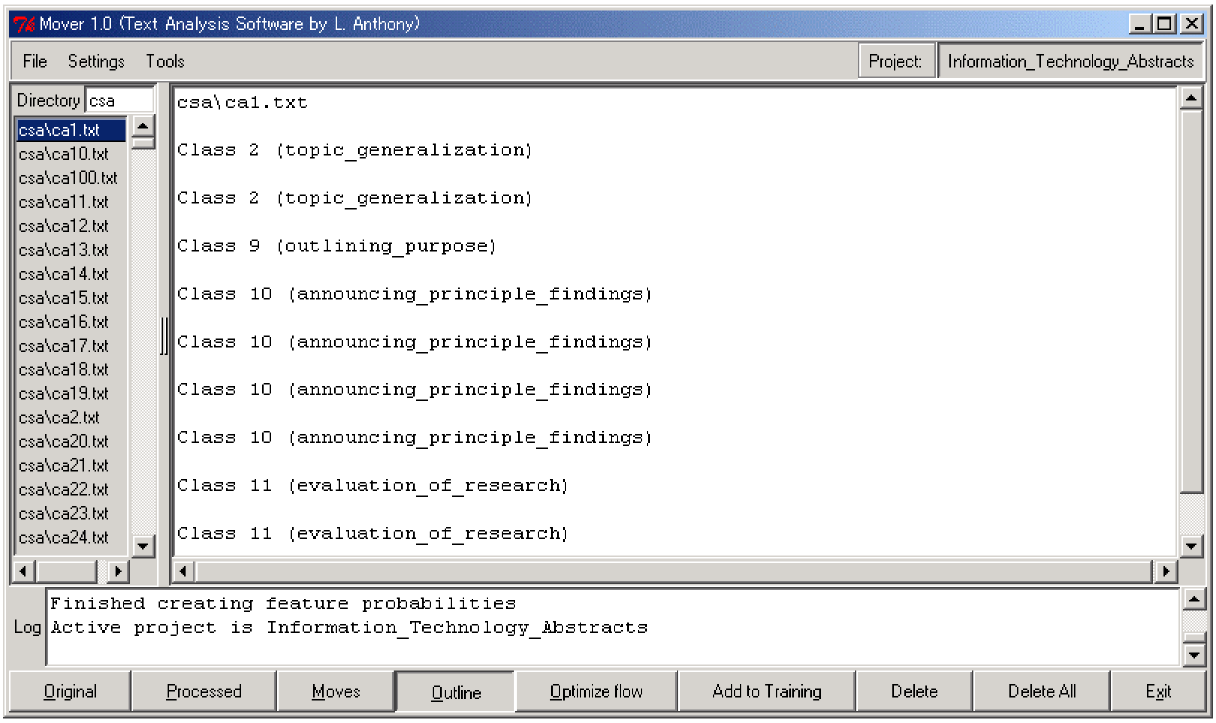
Figure 1.7 *AntMover* in the 'Moves' Viewing Mode



```
csa\ca1.txt

Class 2 (topic_generalization)
The enormous state spaces which must be searched when verifying the correctness of
, or generating tests for , complex circuits precludes the use of traditional
approaches .

Class 2 (topic_generalization)
Hard - to - find abstractions are often required to simplify the circuits and make
the problems tractable .

Class 9 (outlining_purpose)
This paper presents a simple and automatic method to extract the control flow of a
circuit so that the resulting state space can be explored for validation coverage
analysis and automatic test generation .

Class 10 (announcing_principle_findings)
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

Figure 1.8 *AntMover* in the 'Outline' Viewing Mode



```
csa\ca1.txt

Class 2 (topic_generalization)

Class 2 (topic_generalization)

Class 9 (outlining_purpose)

Class 10 (announcing_principle_findings)

Class 10 (announcing_principle_findings)

Class 10 (announcing_principle_findings)

Class 10 (announcing_principle_findings)

Class 11 (evaluation_of_research)

Class 11 (evaluation_of_research)
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

displayed with the class number and label above each step unit of the text (Figure 1.7).

Step 5: Click on the* Outline *button in the 'Button Bar'

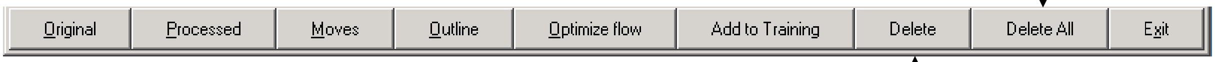
When the *Outline View* is selected, only the class number and label assigned to each step unit of the text is shown. This allows a user to quickly assess the overall structure of the target text (Figure 1.8).

Step 6: Removing files from the 'File List' frame

At any time, the currently selected file or all files in the 'File List' can be removed from *AntMover* (but not erased from the computer) by clicking on either the Delete button or DeleteAll button (Figure 1.9).

Figure 1.9 Deleting Files from the 'File List' Frame

Click here to delete all the files in the 'File List'



Click here to delete the selected file

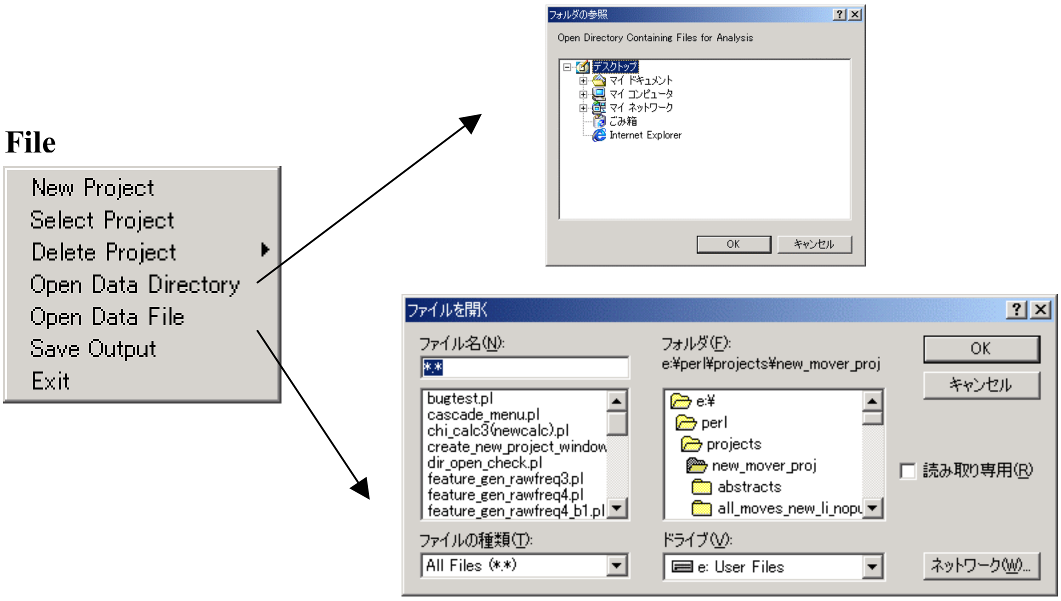
Step 7: Selecting a new file or directory

To open a new file for analysis or a directory of files for analysis, select either the "Open Data File" option or "Open Data Directory" option from the FILE menu. This will bring up a standard Windows dialog box for selecting either files or directories. (Note the language of these dialog boxes will correspond to the language of the OS installed on the user's computer) (Figure 1.10).

Step 8: Analyzing text inputted via the keyboard using the* Line Editor *tool

Text can be input into *AntMover* directly from the keyboard using the *Line Editor* tool. Selecting the *Line Editor* tool from the TOOLS menu, will put an 'active' indicator mark next to the

Figure 1.10 Selecting a file or directory in *AntMover*



Open Directory Containing Files for Analysis

ファイル名(N): *.*

フォルダ(F): e:¥perl¥projects¥new_mover_proj

bugtest.pl
cascade_menu.pl
chi_calc3(newcalc).pl
create_new_project_window
dir_open_check.pl
feature_gen_rawfreq3.pl
feature_gen_rawfreq4.pl
feature_gen_rawfreq4_b1.pl

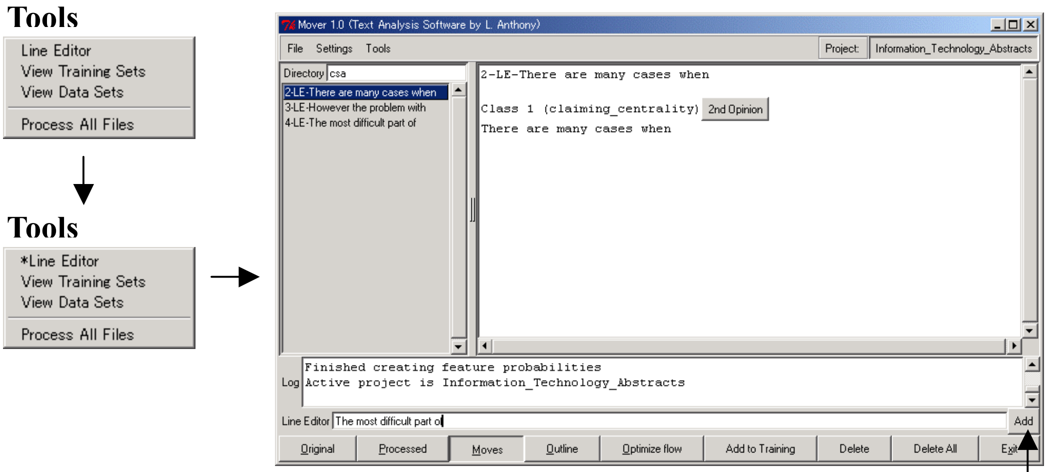
e:¥
perl
projects
new_mover_proj
abstracts
all_moves_new_li_nopu

ファイルの種類(T): All Files (*.*)

ドライブ(V): e: User Files

option, and cause a new 'Line Editor Entry Frame' to appear at the bottom of the 'Main Window'. Text typed into the entry box can be added to the 'File List' frame by pressing the Add button on the right of the frame, or hitting the <RETURN> key. Text added to the 'List Frame' can then be processed in the exactly the same way as a file loaded into the system using the procedure described in Step 7 (Figure 1.11).

Figure 1.11 Inputting Text via the Keyboard using the *Line Editor* Tool



Project: Information_Technology_Abstracts

Directory: csa

2-LE-There are many cases when
3-LE-However the problem with
4-LE-The most difficult part of

```
2-LE-There are many cases when

Class 1 (claiming_centrality)
There are many cases when
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

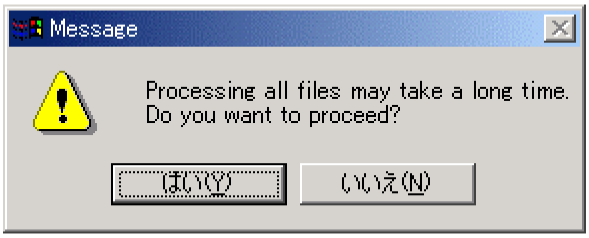
Line Editor: The most difficult part of

Click here to add the line editor text to the 'File List'

Step 9: Batch Processing Files for Analysis

When a file in the 'File List' is selected for the first time, *AntMover* performs all processing and analysis of the file, and displays the results in the chosen view selected in the ‘Button Bar’. For longer texts, this can take several seconds to complete and means a delay occurs before the results appear in the ‘Analysis Frame’. If a large number of files are to be processed and viewed consecutively, it is sometimes preferable to process all the files at once in a single ‘batch’ process. This allows texts to then be viewed in any viewing mode almost instantaneously. To process all texts chose the “Process all Files” option from the TOOLS menu. This will bring up a warning dialog box about the time required to complete the task (Figure 1.12). Press the ‘YES’ button in this box to start the batch processing. Note that all processed files are stored in active memory to allow them to be viewed immediately, so it is important to ensure that enough memory is available to complete the process. Note also that the language of all warning dialog boxes will correspond to the language of the OS installed on the user’s computer.

Figure 1.12 Warning Dialog Box that Appears before Batch Processing of Files

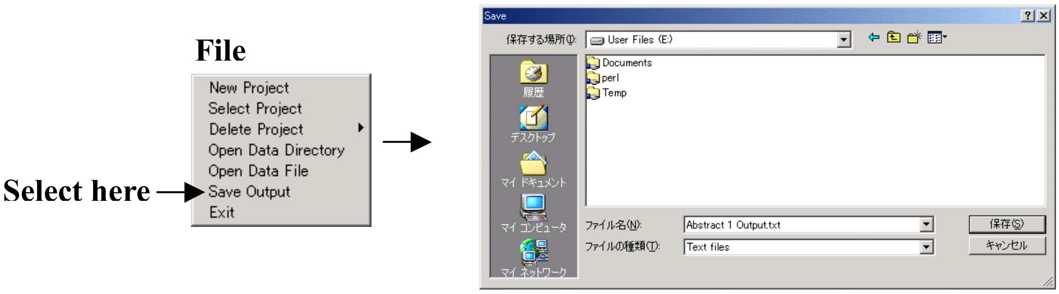


Processing all files may take a long time.
Do you want to proceed?

Step 10: Saving output from the ‘Analysis Frame’

Output that appears in the ‘Analysis Frame’ can be saved at any time by choosing the “Saved Output” option under the FILE menu. This will bring up a standard Windows dialog box to specify the name and location of the file containing the saved output (Figure 1.13).

Figure 1.13 Saving output from the ‘Analysis Frame’



保存する場所(I): User Files (E:)

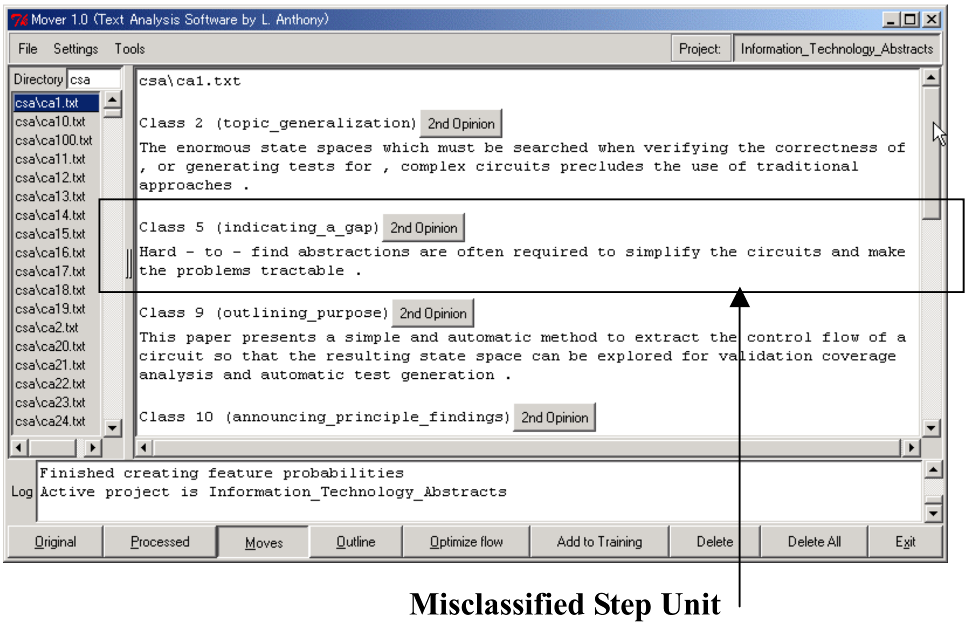
ファイル名(N): Abstract 1 Output.txt

ファイルの種類(T): Text files

1.8 Modifying the Results of the AntMover Analysis

The example in Figure 1.14 shows the results of applying *AntMover* to a chosen text in the 'File List'. From the example, it can be seen that the second step unit of the text has been misclassified by the system. To correct this error, the user has two options available.

Figure 1.14 'Main Window' showing a Misclassified Step Unit



```
csa\ca1.txt

Class 2 (topic_generalization)
The enormous state spaces which must be searched when verifying the correctness of
, or generating tests for , complex circuits precludes the use of traditional
approaches .

Class 5 (indicating_a_gap)
Hard - to - find abstractions are often required to simplify the circuits and make
the problems tractable .

Class 9 (outlining_purpose)
This paper presents a simple and automatic method to extract the control flow of a
circuit so that the resulting state space can be explored for validation coverage
analysis and automatic test generation .

Class 10 (announcing_principle_findings)
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

Misclassified Step Unit

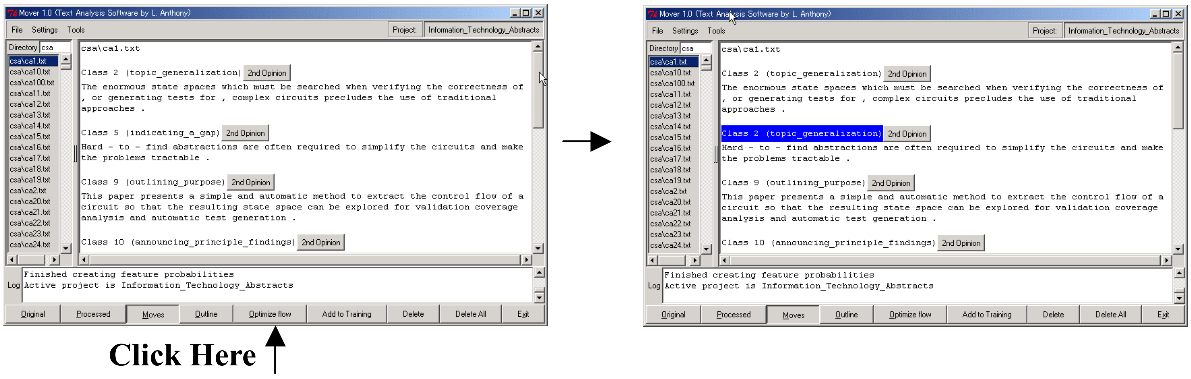
1.8.1. Optimizing the Results of AntMover to Account for Structural Flow

The initial results generated by *AntMover* are based on an analysis of step units in isolation. However, in many cases it is better to consider the 'flow' of the text from one step unit to the next and adjust the results to account for common groupings of structural elements. This adjustment can be performed automatically by clicking on the Optimize Flow button in the 'Button Bar'. If any adjustments are made by the system, they are highlighted in the 'Analysis Frame' in a highlight color. The default highlight color is 'blue' (Figure 1.15).

1.8.2 Getting a Second Opinion about the Results

Sometimes, the system will still make errors even after adjusting for structural flow (see above). In this case, the user can choose to view a ranking of the decisions the

Figure 1.15 Applying Flow Optimization to Correct a Misclassified Step Unit



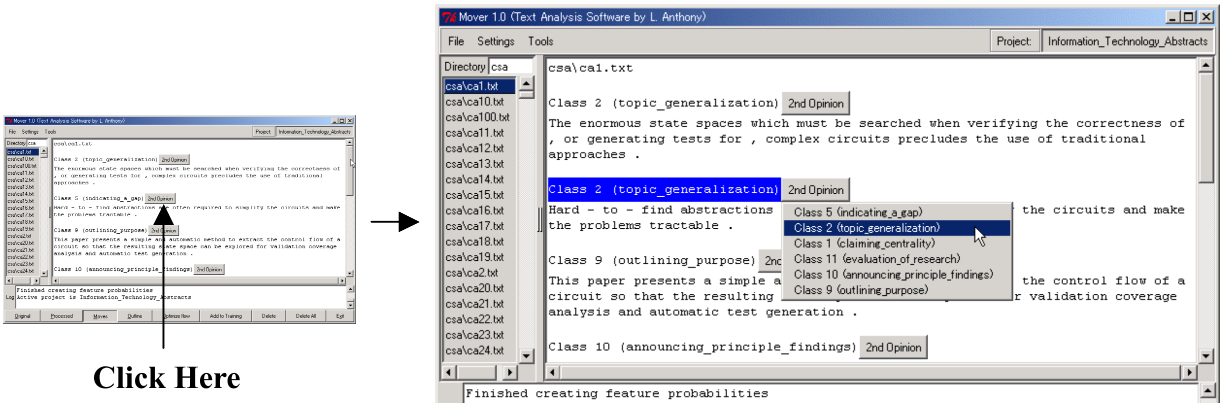
```
csa\ca1.txt

Class 2 (topic_generalization)
The enormous state spaces which must be searched when verifying the correctness of
, or generating tests for , complex circuits precludes the use of traditional
approaches .

Class 5 (indicating_a_gap)
Hard - to - find abstractions are often required to simplify the circuits and make
the problems tractable .

Class 9 (outlining_purpose)
This paper presents a simple and automatic method to extract the control flow of a
circuit so that the resulting state space can be explored for validation coverage
analysis and automatic test generation .

Class 10 (announcing_principle_findings)
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

Click Here

system makes for a particular step unit, by clicking on the 2nd Opinion button in the 'Analysis Frame' next to the step unit label. Clicking on the button brings up a small list box with a ranking of the decisions from the most probable decision at the top to the least probable decision at the bottom. Any ranked decisions can be selected in this list box, and the changes will be highlighted in the 'Analysis Frame' (Figure 1.16).

Figure 1.16 Applying a "2nd Opinion" to Correct a Misclassified Step Unit



```
csa\ca1.txt

Class 2 (topic_generalization)
The enormous state spaces which must be searched when verifying the correctness of
, or generating tests for , complex circuits precludes the use of traditional
approaches .

Class 2 (topic_generalization)
Hard - to - find abstractions                                   the circuits and make
the problems tractable .

Class 9 (outlining_purpose)
This paper presents a simple a                                  the control flow of a
circuit so that the resulting                                  r validation coverage
analysis and automatic test generation .

Class 10 (announcing_principle_findings)
```

```
Finished creating feature probabilities
Active project is Information_Technology_Abstracts
```

Mover 1.0 (Text Analysis Software by L. Anthony)
File Settings Tools
Project: Information_Technology_Abstracts
Directory csa
2nd Opinion
Class 5 (indicating_a_gap)
Class 2 (topic_generalization)
Class 1 (claiming_centrality)
Class 11 (evaluation_of_research)
Class 10 (announcing_principle_findings)
Class 9 (outlining_purpose)
Log
Original
Processed
Moves
Outline
Optimize flow
Add to Training
Delete
Delete All
Exit

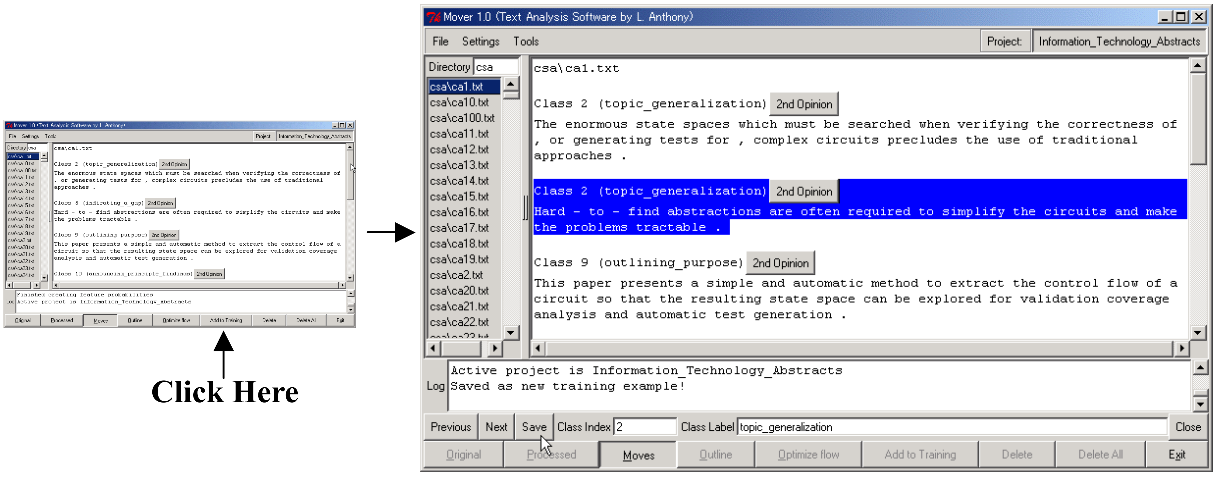
Click Here

1.9 Adding a Processed Text to the Project Training Data

Step units that are misclassified by *AntMover* are very important to the system as a whole. This is because errors relate to a poor internal representation of the target class in the system. By adding these misclassified steps to the training data for a project after they have been corrected, and 're-training' the system on this new data, the system can automatically adjust its internal representation to reflect this new information. This, in turn, results in a greater accuracy of the system over time.

Adding a corrected step unit (or any new step unit) to the training data of a project is easy to carry out. First, correct the label given to the misclassified step unit using one of the methods described in Section 1.8. Then, click on the Add to Training button in the ‘Button Bar’. This creates a new ‘Add to Training’ frame at the bottom of the ‘Main Window’. Using the buttons on the left of the ‘Add to Training’ frame to jump between the different step units in the ‘Analysis Frame’, move the highlighted selection to the misclassified step unit. The class number and label of this step unit is shown on the right side of the ‘Add to Training’ frame. When the right step unit has been selected, press the Save button to save the step unit as a new training example of the specified class (Figure 1.17).

Figure 1.17 Saving a Step Unit as New Training Data



```
csa\ca1.txt

Class 2 (topic_generalization)
The enormous state spaces which must be searched when verifying the correctness of
, or generating tests for , complex circuits precludes the use of traditional
approaches .

Class 2 (topic_generalization)
Hard - to - find abstractions are often required to simplify the circuits and make
the problems tractable .

Class 9 (outlining_purpose)
This paper presents a simple and automatic method to extract the control flow of a
circuit so that the resulting state space can be explored for validation coverage
analysis and automatic test generation .
```

```
Active project is Information_Technology_Abstracts
Saved as new training example!
```

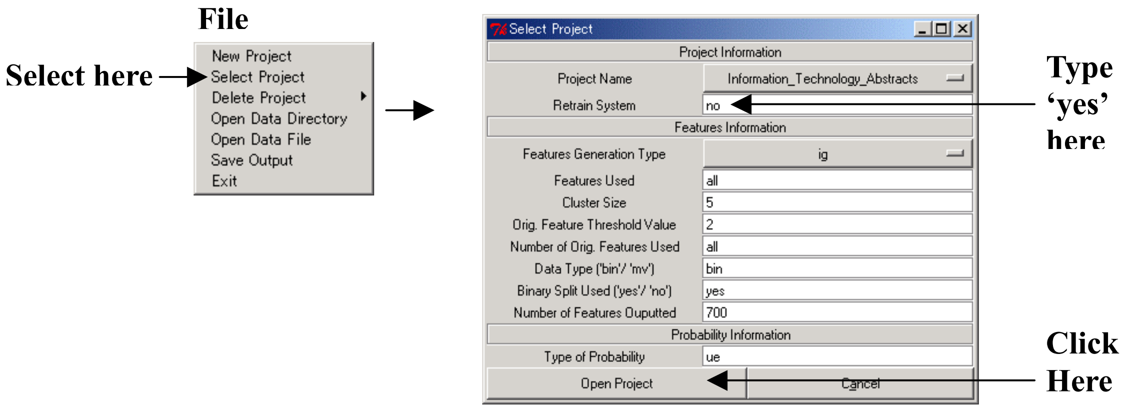
Class Index: 2
Class Label: topic_generalization

Click Here

At any time before pressing the Save button, the step unit labeling can be corrected using either one of the methods described in Section 1.8, or by directly typing in changes in the entry boxes on the right side of the ‘Add to Training’ frame. Note that it is also possible to type new class numbers or labels into the entry boxes, and this will create new classes for the system to learn. This is useful, for example, if a new type of step unit is encountered during the analysis.

After saving one or more new training examples, it is necessary for the system to apply these examples in the creation of a modified knowledge representation of the target class. This is usually termed ‘retraining’ the system. To retrain *AntMover*, choose the “Select Project” option from the FILE menu. This will automatically bring up the current settings of the working project in a ‘Select Project’ window. To retrain the system, enter ‘yes’ in the ‘retraining system' entry box, and click on the Open button at the bottom of the new window to re-open the project after the new knowledge representation has been created (Figure 1.18).

Figure 1.18 Retraining *AntMover* using the Select Project Window



Project Information

Project Name: Information_Technology_Abstracts

Retrain System: no

Features Information

Features Generation Type: ig

Features Used: all

Cluster Size: 5

Orig. Feature Threshold Value: 2

Number of Orig. Features Used: all

Data Type ('bin'/ 'mv'): bin

Binary Split Used ('yes'/ 'no'): yes

Number of Features Ouputted: 700

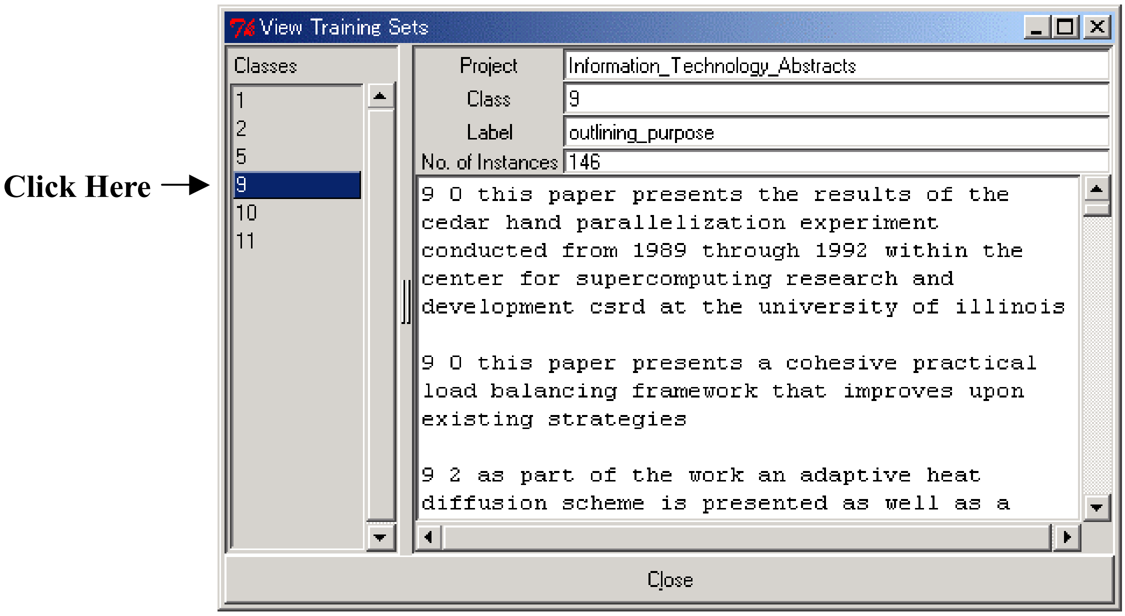
Probability Information

Type of Probability: ue

1.10 Viewing Training Data

All training data used in a particular project can be viewed directly via the 'View Training Sets' window. This can be accessed by selecting the "View Training Sets" option in the TOOLS menu. Clicking on a class in the left hand frame will display the class label, number of examples and an list of all training examples for the class in the right hand frame (Figure 1.19).

Figure 1.19 Viewing Training Data



Project: Information_Technology_Abstracts

Class: 9

Label: outlining_purpose

No. of Instances: 146

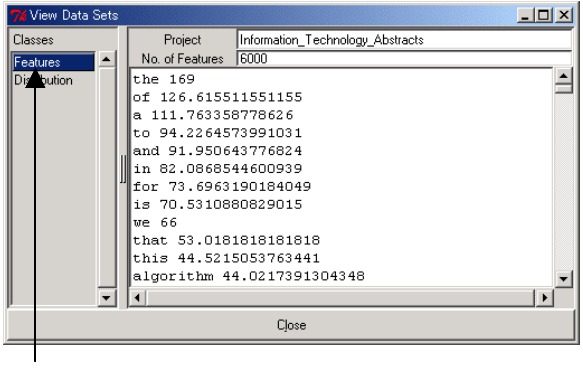
```
9 0 this paper presents the results of the
cedar hand parallelization experiment
conducted from 1989 through 1992 within the
center for supercomputing research and
development csrd at the university of illinois

9 0 this paper presents a cohesive practical
load balancing framework that improves upon
existing strategies

9 2 as part of the work an adaptive heat
diffusion scheme is presented as well as a
```

1.11 Viewing Data Used in a AntMover Project

For each project, a list of all features (clusters) used by the system to represent the individual step units, and the 'bag of clusters' distribution for each training data example can be viewed in the 'View Data Sets' Window. This can be accessed by selecting the "View Data Sets' option in the TOOLS menu. When the 'View Data Sets' window appears, either type of data can be selected in the left frame. Clicking on 'Features' will show the number and listing of features used in a project, ranked according to their score as determined by the feature selection procedure applied. Similarly, clicking on 'Distribution' will show the number of training instances and there feature distributions in the right frame of the window (Figure 1.20).

Figure 1.20 Viewing Data used in a *AntMover* Project

Viewing Features

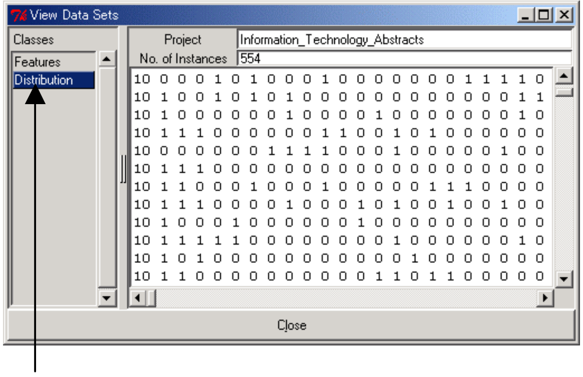


Project: Information_Technology_Abstracts
No. of Features: 6000

```
the 169
of 126.615511551155
a 111.763358778626
to 94.2264573991031
and 91.950643776824
in 82.0868544600939
for 73.6963190184049
is 70.5310880829015
we 66
that 53.0181818181818
this 44.5215053763441
algorithm 44.0217391304348
```

Click Here

Viewing Data Distribution



Project: Information_Technology_Abstracts
No. of Instances: 554

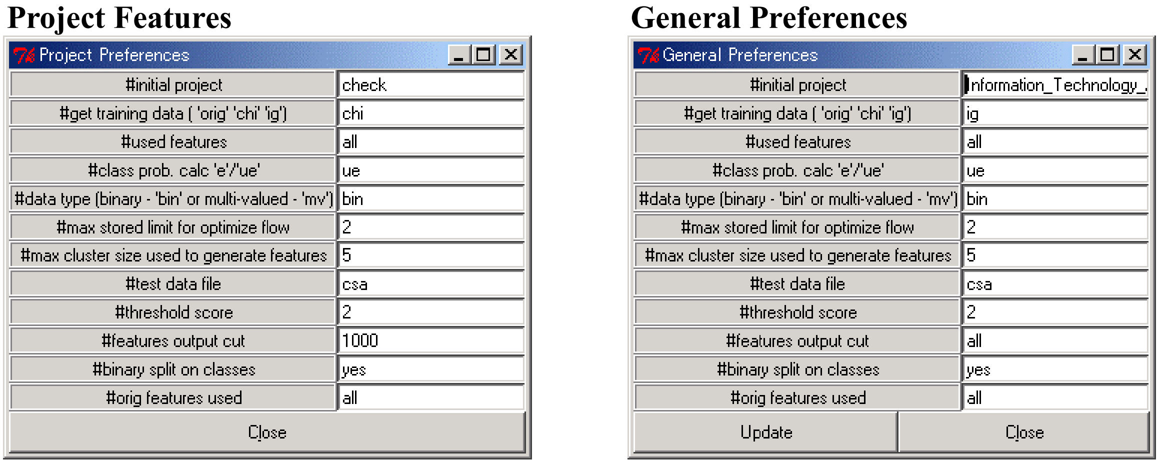
```
10 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 1 1 1 1 0
10 1 0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
10 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0
10 1 1 1 0 0 0 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0
10 0 0 0 0 0 0 1 1 1 1 0 0 0 1 0 0 0 0 0 1 0 0
10 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
10 1 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0
10 1 1 1 0 0 0 0 1 0 0 0 1 0 1 0 0 1 0 0 1 0 0
10 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
10 1 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0
10 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
10 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 0 0
```

Click Here

1.12 Viewing Project and System Preferences

The current settings and preferences in the working project can be viewed in the 'Project Preferences' window that can be accessed by selecting the "View Project Preferences" option in the SETTINGS menu. Similarly, system wide settings that are applied when creating and opening projects can be viewed and altered in the 'General Preferences' window, which is accessed by selecting the "View General Preferences" option in the SETTINGS menu (Figure 1.21).

Figure 1.21 Viewing Project and General Preferences



Project Features

Project Preferences

#initial project	check
#get training data ('orig' 'chi' 'ig')	chi
#used features	all
#class prob. calc 'e'/'ue'	ue
#data type (binary - 'bin' or multi-valued - 'mv')	bin
#max stored limit for optimize flow	2
#max cluster size used to generate features	5
#test data file	csa
#threshold score	2
#features output cut	1000
#binary split on classes	yes
#orig features used	all

General Preferences

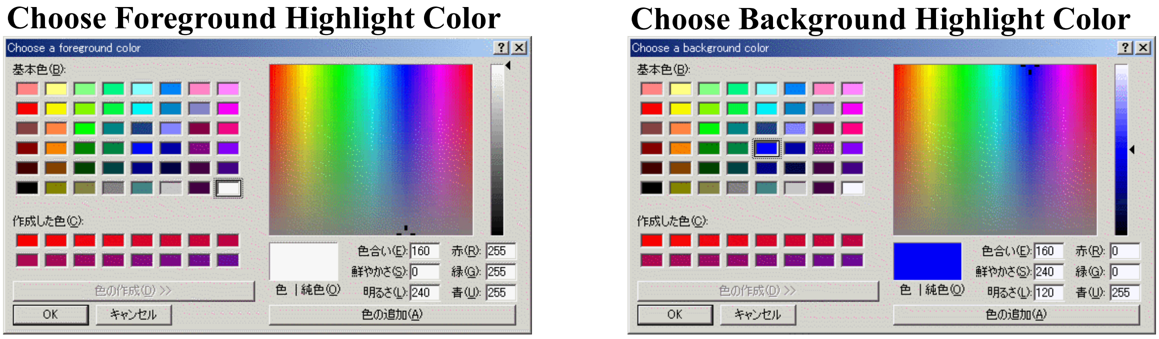
General Preferences

#initial project	Information_Technology_
#get training data ('orig' 'chi' 'ig')	ig
#used features	all
#class prob. calc 'e'/'ue'	ue
#data type (binary - 'bin' or multi-valued - 'mv')	bin
#max stored limit for optimize flow	2
#max cluster size used to generate features	5
#test data file	csa
#threshold score	2
#features output cut	all
#binary split on classes	yes
#orig features used	all

1.13 Changing the Highlight Color

The foreground and background colors used when highlighting a step unit that has been corrected using the 'Flow Optimization' or '2nd Option' features can be changed by selecting the "Choose Highlight Foreground Color" and "Choose Highlight Background Color" options under the SETTINGS menu. This brings up a dialog box in which the desired color can be selected from a wide range of system colors (Figure 1.22).

Figure 1.22 Changing Foreground and Background Color Settings



Choose Foreground Highlight Color

Choose a foreground color

色合い(E): 160 赤(R): 255
鮮やかさ(S): 0 緑(G): 255
明るさ(L): 240 青(U): 255

Choose Background Highlight Color

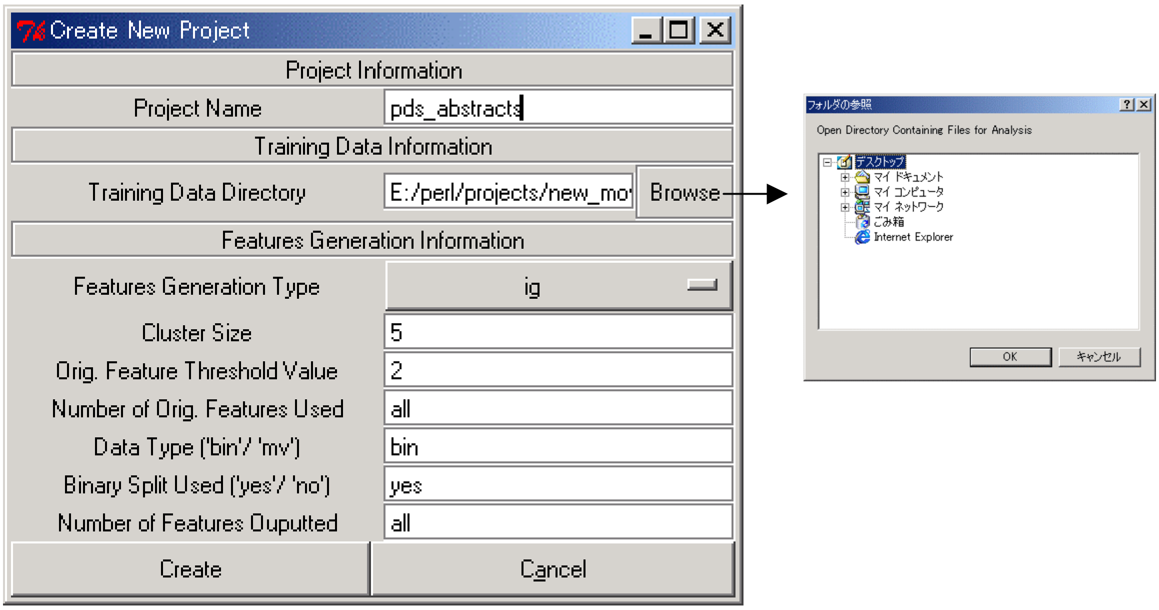
Choose a background color

色合い(E): 160 赤(R): 0
鮮やかさ(S): 240 緑(G): 0
明るさ(L): 120 青(U): 255

1.14 Creating a New Project

Creating a new project within *AntMover* is a relatively simple procedure. First, select the "Create New Project" option under the FILE menu. This brings up the 'Create New Project' window (Figure 1.23).

Figure 1.23 ‘Create New Project’ Window



Project Name	pds_abstracts
Training Data Directory	E:/perl/projects/new_mo
Features Generation Type	ig
Cluster Size	5
Orig. Feature Threshold Value	2
Number of Orig. Features Used	all
Data Type ('bin'/ 'mv')	bin
Binary Split Used ('yes'/ 'no')	yes
Number of Features Ouputted	all

Open Directory Containing Files for Analysis

In the ‘Create New Project’ window, a number of choices have to made. First, type in the name of the new project in the ‘Project Name’ entry box. Next, check the path name of the directory that contains the raw data used for training the system. Note that the default pathway points to a directory “antmover_training_classes” which is created in the “antmover” directory when the system is installed. However, this can be changed by typing in a path name directly, or browsing the OS file system for the directory. Next, select the type of feature selection procedure to be employed. Options available are to select features according to the frequency of occurrence (“orig”), the chi-squared measure (“chi”), or using Information Gain (“ig”).

If the “orig’ option is selected, the following parameters need to be set:

Cluster Size:

This corresponds to the size of the token clusters used in the knowledge representation.

Orig. Feature Threshold Value:

This determines the cut off point at which features will be stored. For example, to only store features that occur with a frequency of two or more select a threshold of two.

Number of Orig. Features Used:

This determines the number of features used in the representation, after the threshold value has been applied. For example, selecting a threshold of two and a number of features of five will force the system to create a representation using only the top five most frequent features that occur with a frequency of two or more in the training data as a whole.

Data Type ('bin' / 'mv')

This determines how the features used by the system are weighted in the 'bag of clusters' representation of training data. The 'bin' option uses a binary (1 0) representation in which a feature has a weight of one if it appears in the training data, and a weight of zero if it is absent. The 'mv' option, on the other hand, weights features by the frequency of occurrence.

To perform CHI and IG feature selection, the system first creates a data representation using the 'orig' option, and uses this to calculate the CHI and IG scores depending on which has been selected. Therefore, all the settings for the 'orig' feature selection option also apply to the 'chi' and 'ig' options. In addition, two additional parameter settings are required if the 'chi' and 'ig' options are selected.

Binary Split Used ('yes' / 'no')

This parameter determines whether or not the CHI or IG calculation is treated as a multiple class problem, or a set of binary problems, whereby the target class is treated as one class and the remaining classes are combined and treated as a second class. Note, this option creates a large number of features if many classes are considered.

Number of Features Outputted

This parameter determines how many features are generated using the CHI or IG measures, regardless of how many original features are generated. Clearly, if a binary split option is not used, the maximum number of features that can be outputted will equal the number of original features. (Using 'all' defaults to the maximum number of features possible). On the other hand, if a binary split option is applied and all features are used, the resulting number of generated features will equal the original number of features multiplied by the number of classes.

The default settings in the 'Create Project' window can be set in the 'General Preferences' window described above.

When the Create button is pressed in the 'Create New Project' window, *AntMover* will load in the raw training data from the directory specified, and convert the files into a specialized format that is understood by the system. The converted files are then stored in the 'Training Data' directory under the newly created project directory. Following this, the training files are processed and the new project knowledge representation is created. To ensure that the conversion process is successful, the raw training data has to be organized according to the structure given in Figure 1.24.

Figure 1.24 Directory Organization for Raw Training Data

```
raw training data directory
          (class number)_(class label)
                   abc.txt
                   def.txt
                   ghi.txt
                   …
          (class number)_(class label)
                   jkl.txt
                   mno.txt
                   pqr.txt
                   …
```

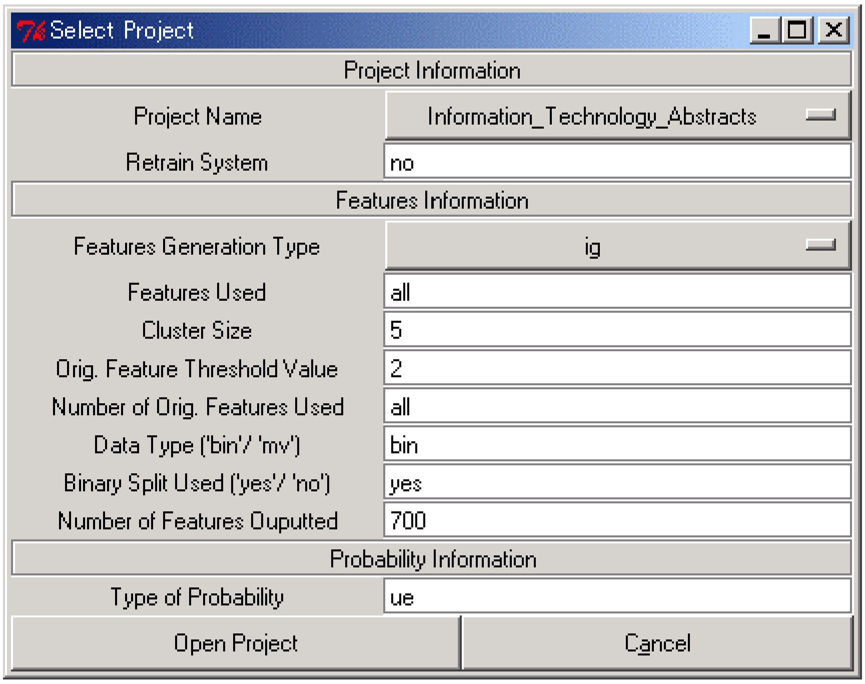
From Figure 1.24, it can be seen that each class must be correctly named with a class number followed by an underscore, followed by the class name (no spaces included).

1.14 Opening an Existing Project

One important feature of *AntMover* is that it can be used to investigate the structure of a target text using information stored in any project generated by the system. The active project can also be switched at any time, allowing a user to see the effects of applying different structural models and parameter settings immediately. This can be useful, for example, if the target text directory contains a multitude of different text types from a wide range of disciplines, or for investigating which structural model from a set of models in the system applies best to a corpus of data.

To select a project, choose the "Select Project" option under the FILE menu. This opens the 'Select Project' window (Figure 1.25). The various parameter settings that were

Figure 1.25 The ‘Select Project’ Window



Select Project

Project Information

Project Name	Information_Technology_Abstracts
Retrain System	no

Features Information

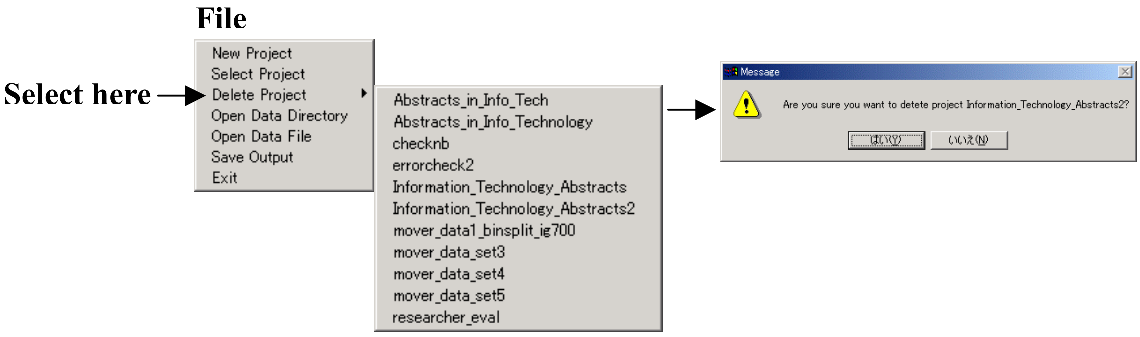
Features Generation Type	ig
Features Used	all
Cluster Size	5
Orig. Feature Threshold Value	2
Number of Orig. Features Used	all
Data Type ('bin'/ 'mv')	bin
Binary Split Used ('yes'/ 'no')	yes
Number of Features Ouputted	700

Probability Information

Type of Probability	ue

used to create the project are displayed in a similar way to those in the ‘Create Project’ window. In addition, the ‘Select Project’ window can be used to retrain the system after new training data has been added to a project via the ‘Add to Training’ tool described earlier. Also, because *AntMover* uses a Naïve Bayes classifier to determine the class of a target text, it is possible to set the classifier to work with even class probabilities (e) or uneven class probabilities (ue). Using the (e) setting means that the system will treat all classes as equally probable when determining the class of a target text. In contrast, the (ue) setting will cause the system to base class probabilities on their distribution within the training data.

Figure 1.26 Deleting a Project in *AntMover*



Message

Are you sure you want to detete project Information_Technology_Abstracts2?

1.15 Deleting an Existing Project

If a project is no longer required in *AntMover*, it can be deleted (erased from the system) using the “Delete Project” option under the FILE menu. After selecting the project to be deleted, a warning dialog will appear to confirm the action about to be performed. If the user chooses ‘YES’ in this dialog box, the project will then be deleted (Figure 1.26). Note that the active project cannot be deleted.