Small Expert System - Main Index

Software

Background

Dialogue Boxes

Miscellaneous

Tips and Troubleshooting Guide

1. This version is bound to have a few hiccups which will be removed from future versions. If you find that you can not do anything or if the program *appears* to lock up, that means that most likely the expert system engine is waiting for a response and for some reason all the dialogue boxes have been closed. In this case the best thing to do is to try the **Quit** option under the main **File** menu. This will cause the program to disentangle itself from the expert engine, and you will be able to continue.

2. Always save your work as often as possible to minimize the chances of a horrible disaster (the night before a project is due, for example).

3. Click on the **Ask** button to bring the **Question** or **Decision Reached** dialogue box to the forefront if your desktop is cluttered with millions of windows.

How To Model It

This expert system shell accompanies Chapter 10 of the book **How to Model It: Problem-solving for the Computer Age** by Tony Starfield, Karl Smith and Andrew Bleloch.

In an active learning environment, each chapter of the book creates a context for modeling, poses a problem, and guides the reader through the process of problem definition, solution development, strategy, model building and interpretation of modeling results. The book is aimed at readers with little or no prior modeling experience. It has been used by high school students, in both undergraduate and graduate classes in engineering, business, science and education, and also in professional development workshops.

Jerry Pournelle, BYTE Magazine, March 1991 copyright McGraw-Hill, Inc wrote of this book: ..... *it is a serious attempt to teach modeling..... it's the best I've seen on the subject.*

How to Model It is available from Burgess International Group, Inc, 7110 Ohms Lane, Edina MN 55439-2143. Their toll free phone number is 800/356-6826 and their fax number is 612/831-3167.

The authors of this book cannot accept any legal liability for any damage or losses incurred as a result of using this software.

Future Updates

This version of the software is **version 1.0**.

As a first version, there are bound to be bugs, and problems (some of which we already know about) which would be fixed in future versions of the software. The developers of this software cannot accept any legal liability for any damage or losses incurred as a result of using this software.

All suggestions and comments are welcome, and can be sent to the developers of the software via email to the authors of the book How to Model It (internet):

Dr. A.M.Starfield: starf001@maroon.tc.umn.edu
Dr. K.A. Smith: ksmith@vx.cis.umn.edu

or by snail mail to the developers care of Dr. A.M.Starfield:

Dr. AM.Starfield
1614 Rosehill Circle
Lauderdale
MN 55108
USA

The expert system engine has a lot of logic built into it specifically looking for bugs which we have not yet found...(and hopefully will never find!). These check points which slow the engine down will be removed from future versions.

This version also makes use of temporary files, simply because it was the easy way out. The **trail** feature, for example, uses a temporary file ***trail.$$$.*** This will cause problems if more than one instance of the **Small Expert System** is running, and also slows the program down quite a bit. Future versions will not use temporary files in this manner.

We will be making improved versions available as Shareware programs some time in the future. These will be more efficient and will have additional features such as:

A **Generate** option to generate a simple C program which runs your particular knowledge base, but which can be incorporated into the source code of another application so that the expert system engine becomes an integral part of that application.

At least two additional optional optimizations.

Improved formatting of text, and improved printing and exporting.

Builder

Printing and Exporting

The entire database can be displayed using the **Display All** option, and can also be printed from the **Display All** window. This version of the software does not attempt to format the output or make it pretty. So if you need to print out the actual knowledge base, then use the **Export** option under the main **File** menu to export a text file which you can import into a word processor or editor and reformat. The printing and formatting will be improved in a future version.

Note that the expert system knowledge bases developed on this system are saved in a binary format that will not make sense to any other program.

Context

Type in anything that is relevant to this knowledge base. This is not used by the expert system. The context is the first thing that pops up on the screen when a user loads a knowledge base.

Decisions

Use the **Next** and **Previous** buttons to look through the decisions. If you need to jump to a decision directly, enter the decision number and press the **Display Decision** checkbox. Adding and deleting decisions are done by means of the **add** and **delete** buttons.

Note: when you delete a decision, all the decisions after it are renumbered so that there is not a gap in the sequence of decisions. For example, if you delete **Decision 5**, then **Decision 6** will be re-labeled as **Decision 5**, and so on. The **Builder** will automatically go through any rules that you have already created and will re-label all references to decisions, but will NOT re-label the reference to **D5**.

The explanation for the decision is entered here.

Questions

Use the **Next** and **Previous** buttons to look through the questions. If you need to jump to a question directly, enter the question number and press the **Display Question** checkbox. Adding and deleting questions are done by means of the **add** and **delete** buttons, and answers to each question are added and deleted in the same way.

Note: when you delete a question, all the questions after it are renumbered so that there is not a gap in the sequence of questions. For example, if you delete **Question 5**, then **Question 6** will be re-labeled as **Question 5**, and so on. The **Builder** will automatically go through any rules that you have already created and will re-label all references to questions, but will NOT re-label the reference to **Q5**.

You can modify an answer by double clicking on it.

The explanation for the question is entered in this dialogue box.

Rules

Use the **Next** and **Previous** buttons to look through the rules. If you need to jump to a rule directly, enter the rule number and press the **Display Rule** checkbox. Adding and deleting rules are done by means of the **add** and **delete** buttons. In the decision field, enter a number, not a decision. For example, enter **1** instead of **d1**.

Question

Choose an answer to the question and click on it. Then press **<Enter>** or the **OK** button. Pressing the **Abort** button causes the expert system engine to quit immediately. Pressing the **Why** button opens the explanation window.

Decision reached

Press the **OK** button if you wish to continue searching for more decisions, otherwise pressing the **Cancel** button causes the expert system engine to quit immediately. Pressing the **Why** button opens the explanation window.

Trail

The **Trail** dialogue box gives all the information about the questions asked and decisions reached that the expert system engine knows about. If a question is asked, the **Trail** window shows both the text of the question asked, and the text of the answer chosen by the user. If a decision is reached, the **Trail** window displays the text of the decision reached. If the engine realizes that a certain decision cannot possibly be reached, the **Trail** window displays a message indicating that the decision has NOT been reached.

Every time the **Trail** changes, the background colour of the window changes from white to grey, and the user needs to press the Update menu option in order that the **Trail** window be up to date. This usually happens after a question has been answered, or a decision has been reached. The **Trail** window will also display a report message or an error message after the engine has quit.

If you are running multiple instances of the **Small Expert System**, you will run into problems with the trail which will be resolved in future updates.

Why

The Why window is opened whenever the **Why** button is pressed in either the **Question** dialogue box or the **Decision Reached** dialogue box.

If the **Why** window is opened from a **Question** dialogue box, then many possible explanations may appear in the window. The first is the explanation on the question asked (which appears under that question in the **Builder** dialogue box), and all subsequent explanations are associated with the rules which are being considered by the engine in order to attempt to reach a particular decision. These latter explanations are the ones which appear under the rules in the **Builder** dialogue box.

If the **Why** window is opened from a **Decision Reached** dialogue box, then the explanation given is the explanation associated with the decision that has been reached.

Description of Engine

What is the engine?

The expert system engine is the part of the program that actually prompts the user to answer questions, and informs the user when a decision has been reached, i.e., the part of the program that runs when the user presses the **Ask** button, or selects **Query** under the **Run** menu. The knowledge base (which the user has either loaded or has typed in) is parsed into a form which the engine can use. This expert system engine is capable of reaching many decisions, and is designed primarily for small knowledge bases.

Where does the engine start?

The engine searches through the rules, trying to first prove decision 1. If it comes across a rule which ends with a **THEN d1**, it evaluates the conditional part of the rule. If the condition is evaluated as **true**, then the decision is marked as **true**, and all other rules which end with a **THEN d1** are ignored. If the condition is evaluated as **false**, then the engine looks for additional rules which end with a **THEN d1**. The decision will only be marked as **false** if ALL the conditional parts of the rules which end with **THEN d1** are evaluated as **false**.

What happens with decision references?

What happens when there is a decision reference in the if part of the rule?

In this case, the engine will attempt to evaluate the decision that appears in the condition. For example:

if (d2 and q2a3) then d1

If the engine needs to know what the value of *d2* is in order to evaluate the Boolean expression, then it will try to prove that decision (as described in the previous section). If the condition looks like:

if (not d2 and q2a3) then d1

Then *(not d2)* will only evaluate as **true** if *d2* evaluates as **false**, and as described in the previous section, this will only happen if the conditional part of every single rule that ends with **THEN d2** is evaluated as **false**. As can be imagined, this backward chaining means that there is the possibility of a circular argument, such as the following:

if d1 then d2
if d2 then d1

The engine will detect circular arguments such as these, and will quit with an error message (hopefully!).

How does the engine evaluate conditional expressions?

If the expression is simple such as:

q1a2 and q3a1 and q2a4

then in this case, the engine will start at the left, and will first ask the question q1 (if it has not already been asked), followed by *q3* and *q2*, in that order. Do not mix **ORs** and **ANDs**, but rather use parentheses to establish which operator has precedence. If the expression is more complicated, such as:

(q1a2 and q3a2) or ((not q2a3 and q4a2) or q5a1)

then in this case which question is asked first depends on a few factors. If any of the questions have already been asked, then the engine will try to make sure it asks as few questions as possible. For example, if *q4a2* is already known to be **false**, then there is no point in asking *q2a3*, because no matter what the answer to question 2 is, the expression (not *q2a3* and *q4a2*) is going to be evaluated as **false**. In the same way, if *q5a1* is already known to be **true**, then there is no point in asking either question 2 or question 4, because the expression *((not q2a3 and q4a2) or q5a1)* is going to be evaluated as **true** anyway.

For this more complicated expression, the engine sees something like:

A or B

where

A = (q1a2 and q3a2)
B = ((not q2a3 and q4a2) or q5a1)

but will NOT automatically start at the left and evaluate *A*, and then *B*. The engine checks to see how many questions it would need to ask (in the worst case scenario) to evaluate *A*, and if *B* would require more questions, then *A* will be evaluated first. Exactly how the engine determines which expression to ask first depends on the whether or not the short-circuit search optimization has been turned on (by default, it is ***on***). If this optimization is not on, then the engine will simply count how many unanswered questions there are in *A* and *B*, and will evaluate *B* first if *B* has fewer unanswered questions. For example, in the above example, if all the questions have not yet been asked, then *A* will be evaluated first because it has two unanswered

questions, as compared with three unanswered questions in *B*. If *A* is evaluated first and is found to be true, then *B* will not be evaluated, and similarly if *B* is evaluated first. If, for example, *q2a3* is known to be true, but all the other questions have not been asked, then whether or not the short-circuit search optimization is on will determine whether *A* or *B* is evaluated first. If this optimization is off, then the engine will count two unanswered questions in *A*, and two unanswered questions in *B*, and consequently will evaluate expression *A* first. However, if **the short-circuit optimization** is on, then because the expression *((not q2a3) and q4a2)* will be evaluated as false no matter what the answer to question 4 is, the engine will only count one unanswered question in *B*, and so *B* will be evaluated first. By default, the short-circuit optimization is on. If there are references to decisions in the logical expression, then the situation becomes even more complicated.

Which expression is evaluated first?

How does the engine decide which expression to evaluate first if there are references to decisions?

If a conditional expression contains references to a decision, such as in the following example:

(q1a2 and d2) or ((not q2a3 and q4a2) or q5a1)

then exactly how the engine decides whether to evaluate *A* first or *B* (the above expression is *(A or B),* as before) depends on whether the decision-depth optimization has been turned on. By default, this optimization is OFF. If this optimization is off, then *d2* in the above expression is treated exactly the same as *q1a2*, i.e., if the decision 2 is not known to be either true or false, then it counts as one unknown. So if all the questions have not yet been asked, and *d2* is not known to be either true or false, then the left hand side will be evaluated first, because it has two unknowns as opposed to three unknowns in the right hand side. Thus the left hand side will be evaluated first no matter how many questions would need to be asked in order for *d2* to be evaluated.

If the decision-depth optimization is switched on, then the engine will look more closely at *d2* and will count how many questions would need to be asked in order to evaluate *d2* (in the worst case scenario) while it decides which half of the expression to evaluate first. So if two questions would be needed to be asked in order to determine *d2*, then the right hand side of the expression would be evaluated first. By default, the decision-depth optimization is OFF. The two optimizations are independent of each other.

Should the optimizations be used?

Should the optimizations be used or not?

This depends on the user. If either or both of the optimizations are off, the engine runs more quickly, but is likely to ask more questions than is necessary in order to reach the same decisions. However, if the knowledge base has been well structured and is simple, then it is possible (although not likely) for the engine to run quicker AND ask fewer questions than if the optimizations were turned on, but this requires a well thought out and structured database that has been specifically designed for an expert system shell which always evaluates expressions from left to right, for example. Turning the optimizations off gives more control as to which questions are asked first, and it also gives more control as to the order in which decisions are reached. By default, the short circuit search optimization is ON and the decision depth optimization is OFF, in accordance with the wishes of the authors of How To Model It, who have had extensive teaching experience with a small expert system shell such as this one.

Getting Started

As an example, click on the first button on the left on the tool bar to load a file, or go the **File** menu, and select **Load.** Select ***burn.kb***.The Builder automatically appears on the screen, with the context displayed. Click on the **decisions** button at the top of the Builder dialogue box, and browse though the decisions using the **Next** and **Previous** buttons. Do the same for the **Questions** and **Rules**. Now press the **Ask** button on the tool bar. A question dialogue box appears with a question, and a choice of answers. Click on the **Why** button. This opens a window which gives you an explanation as to why that particular question is being asked. Next, click on the **Trail** button on the tool bar. This opens another window, which gives a summary of what the expert system engine knows so far. Choose an answer in the question dialogue box by clicking on one of the choices, and then press the **Ok** button or hit the <**enter**> key. The trail window changes colour, so that means you have to select the **update** option if you want to see the updated trail. At any stage, pressing the **Help** button will bring up a relevant help page.