DEB: A Diagnostic Experience Browser using Similarity Networks

Cyprian E. Casadaban

Martin Marietta Manned Space Systems
Post Office Box 29304 Mail Stop 3691
New Orleans, Louisiana 70189

Abstract

This paper describes DEB: a fusion of knowledge base and data base that allows users to examine only the data which is most useful to them. The system combines a data base of historical cases of diagnostic trouble-shooting experience with similarity networks. A menu-driven natural language interface receives input about the user's current problem. Similarity networks provide the user with references to past cases that are most similar or most related to those they now face. The user can then choose the case that is most pertinent and browse its full textual description which, in turn, may include references to other related cases.

Introduction

The following describes some preliminary results of a NASA Mission Task being performed by the Machine Intelligence Group at Martin Marietta Manned Space Systems in New Orleans, where the External Tank for the Space Shuttle is assembled. The goal of the project is to increase productivity at weld stations by decreasing downtime (Pulaski/ Casadaban 88).

When a downtime is reported a team of weld experts responds to the call. They work together to determine the cause of the problem. Their goal is to find out how to get the weld station operational as soon as possible and what to do to keep the problem from recurring. The result of a weld team call is a completed form that explains the path problem solving took from initial diagnosis through solution.

DEB grew out of a need to use this weld downtime trouble-shooting information that is gathered for each occurrence but not referenced afterward. Consequently, when a problem arises that is very similar to a previous problem, the diagnosis process must start from scratch. Instead, knowledge stored about similar downtime episodes could be consulted to lend advice about what to investigate first.

Once it was decided that the domain of aerospace hardware welding was too broad for a conventional expert system, an effort was started to create a history-based trouble-shooting assistant.

Data and Categorical Knowledge

The weld trouble-shooting team originally kept the historical data base in a limited form. A page or two of trouble-shooting information was recorded on paper but only the bare minimum of whom, what, and when was stored in the data base.

Our first task was to see that all data recorded at a weld downtime occurrence not only be written down, but typed into a dBase III plus data base as well (Figure 1). In this way all knowledge of an event was now retrievable. Data capture began in January of 1988.

Case_Number	code distinguishing each case. EX: 88/100 is the 100th case in 1988.
Tool_Number	tool on which the downtime occurred.
Date_of_Occurrence	date on which the downtime occurred.
Hours_Down	# of hours the tool was inoperable before the problem was fixed.
Vehicle_Effectivity	number of the assembly in the weld fixture EX: LW-51
Problem_Component	main component cited in the problem. EX: ROUTER
Problem_Subcomponent	subpart of the component cited in the problem. EX: MOTOR
Problem_Action	action cited in the problem. EX: NOT-OPERATING
Cause_Component	main component cited in the cause. EX: LIMIT-SWITCH
Cause_Subcomponent	subpart of the component cited in the cause EX: ARM
Cause_Action	action cited in the cause. EX: NOT-CONTACTING
Problem_Text	textual description of the problem.
Cause_Text	textual description of what the cause of the problem was.
Action_Text	textual description of what actions were taken for the fix.
Follow_Up_Text	any action items that will keep the problem from recurring.

Figure 1. The Weld Downtime Data Base Structure.

Once all the information about a downtime episode is collected, the case is categorized. This categorical knowledge lies at the heart of DEB and consists of a problem and a cause generalization.

Each generalization breaks down further using a BNF grammar to yield a systematic breakdown of allowable feature values for a given field. The problem and cause are described according to the following feature triplet:

COMPONENT / SUBCOMPONENT / ACTION

Our local expert assigns a category for the problem and cause of each case. For example, suppose the problem at a certain downtime incident is:

"Router motor is down at tool T01A5103"

The expert might categorize this problem as follows:

ROUTER / MOTOR / NOT-OPERATING

The cause of the downtime incident is categorized similarly.

Once the data is captured, it allows for the creation of informative data base sorts. Reports can then be generated showing which problems are occurring at particular tools, when, and what is causing them.

System Description

DEB, in essence, receives user input about a case and finds the top ten most similar cases. This happens in several steps. First, a C program pre-processes dBase III plus data records into the format of KnowledgePro topics. Topics are the data structure for representing similarity network knowledge. Next, these topics are read and used as frames depicting each weld downtime case, component, subcomponent, and action category used, as well as each weld tool. The user is prompted to describe their case by selecting the date of occurrence, the tool, and the component, subcomponent, action triplet which corresponds to the symptoms of their problem.

DEB then uses these parameters to find the ten most similar cases to the current problem. This is accomplished using similarity networks (Bailey 88) wherein a similarity value is retrieved for each aspect of a case as compared to the corresponding aspect of another case. These aspects are component, subcomponent, action, weld-gantry-type, weld-tool-type, weld-type, and recency of occurrence. This knowledge resides in the C-program-generated topic frames. The top ten most similar problems are derived and presented in a menu where the user can select a particular case and browse it. When a case is selected, another C program takes over and locates the case in a dBase III plus file, formats the contents of the case and sends it back to KnowledgePro in a text file which is then displayed for the user.

The system is written in KnowledgePro from Knowledge Garden Inc., and Turbo C from Borland International. Data and categorical knowledge are stored in a dBase III plus data base and similarity network knowledge is stored as KnowledgePro topics used as frames. A similarity network is a simple knowledge representation scheme which can be thought of as a set of objects bound by weighted links.The DEB system configuration is shown in Figure 2.

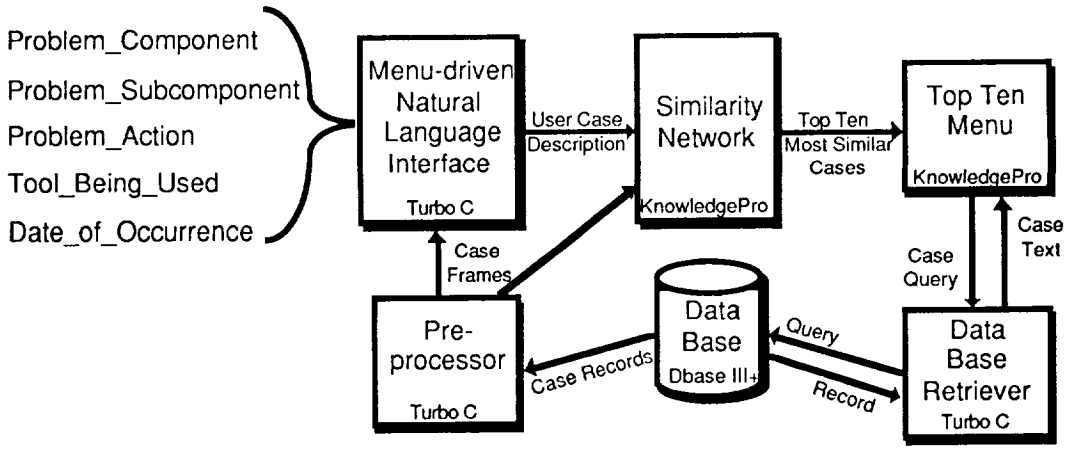

Figure 2. DEB System Configuration.

KnowledgePro was chosen for its ability to control hypertext with an underlying language, a rich data structure (topics), and a backward chaining inference engine which will be used for a future extension. KnowledgePro proved to be extremely slow communicating with the data base, so a faster data base retrieval system was written in Turbo C.

How It Works

First, the pre-processor systematically rummages through the data base and creates a summary frame for each case description (Figure 3). A frame is also created for each unique category. This feature frame contains information about the cases this particular feature has occurred in, which features it is similar to, and which other features it has been associated with in past case symptom triplets. In this way the pre-processor creates a similarity network for each problem feature.

```
Case 88/197
     case_number          88/197
     tool_number          T01A5002
     record_number        59
     date_of_occurrence   06/21/88
     problem_triplet      (ROUTER MOTOR NOT-OPERATING)
     cause_triplet        (ROUTER MOTOR DEFECTIVE)
end

component ROUTER
     is_similar_to        ((SAW 0.8))
     cases_where_seen     (88/197 88/209 88/263)
     subcomponent_list    (MOTOR HANDLE MUFFLER INCORRECT-INSTALLATION)
end
```

Figure 3. Case Summary and Component Feature Frame Examples.

Next, processing the degree of similarity of one feature to another is handled by C functions and is procedural (speed constraints on the delivery product required that this part be rewritten in C and integrated with KnowledgePro, after the intial KnowledgePro prototype). These similarity values reflect how similar in function or how related two problem features are.

At the start of the project our expert was consulted and supplied us with similarity measures in the form of fuzzy linguistic comparisons for all problem features (Schmucker 84). These measures of similarity or relatedness can be visualized as a network. The nodes are features being compared and the arcs are the results of the comparison (Figure 4). Arcs representing totally-different are not shown.

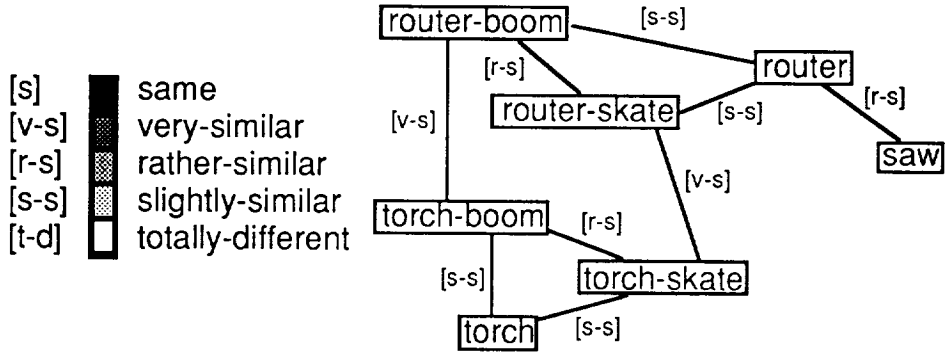

Figure 4. Component-feature Similarity Network Fragment.

Each problem feature type has a static weight associated with it based on its importance in the comparison of two cases (Figure 5). These weights were derived from interviews with our expert.

Once the similarities for each feature are found they are converted into numbers (same: 1.0, very-similar: 0.8, rather-similar: 0.6, somewhat-similar: 0.4, slightly-similar: 0.2, totally-different: 0.0). These numbers are then multiplied by the appropriate feature weight. In this way the total similarity of a similar case versus the base-case will fall in the range of 0.0 to 1.0, where the greater value reflects a greater similarity.

(0.6) Symptom-triplet Feature
- (0.6) Component
- (0.2) Subcomponent
- (0.2) Action

(0.3) Tool Feature
- (0.7) Weld-gantry-type
- (0.2) Weld-tool-type
- (0.1) Weld-type

(0.1) Date Feature or Recency

= 1.0

Figure 5. Feature Weights.

The weighted similarity value is the measure of how similar a feature is to the base feature, multiplied by the appropriate feature weight.

The way the networks are processed is as follows:

1. The user provides the system with a description of the problem (i.e., date, tool, component, subcomponent, action). This is called the *base-case*.
2. The frame corresponding to the *base-component* is accessed. The cases in which this component has been used are stored in the *similar-case-list* with a weighted *similarity-value* of "same" (since these cases have this component in common).
3. Any entries from the *is_similar_to* frame slot are stored as *similar-feature-entries* along with their weighted *similarity-values*.
4. For each *similar-feature-entry* (if any), access its frame and perform step 2 only, appending its case list to the *similar-case-list* along with the weighted *base-component-to-similar-feature-entry-similarity-value* found.
5. Repeat steps 2, 3, and 4 for the *base-subcomponent* and *base-action* just as for the *base-component* all the while appending to the *similar-case-list* or accumulating the weighted similarity of a *similar-case* to the *base-case* if that *similar-case* is already on the *similar-case-list*.
6. For the *base-case*, access the frame corresponding to its tool and find the tool's *weld-gantry-type*, *weld-tool-type*, and *weld-type* features (Figure 6).
7. For each case on the *similar-case-list* perform steps 8, 9, 10 & 11.
8. Access the frame corresponding to the case's tool and obtain the tool's *weld-gantry-type*, *weld-tool-type*, and *weld-type* features.
9. Access the frames for each *tool-feature* and store its weighted similarity measure against the appropriate *base-case-tool-feature*.
10. Compare the *similar-case-date* to the *base-case-date* and calculate *recency*(weighted) according to a formula in which a case is rated higher when it is more recent.
11. Calculate overall similarity versus the *base-case*.

12. Sort the *similar-case-list* by overall similarity and show the user a menu of the top ten most similar cases (Figure 7).
13. The user can now choose a case to browse. Once their selection is made, the full case description is displayed and related cases can be browsed.

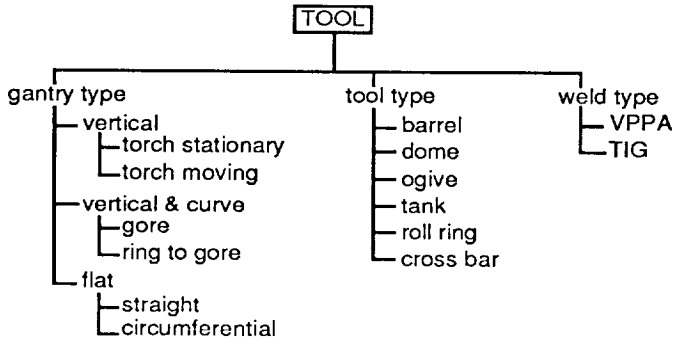

Figure 6. Weld Tool Fearures Hierarchy.

Interface Description

The DEB interface uses both menu-driven natural language and hypertext, which insures that every selection the user can make is valid. Both the initial user interface and the top ten browser were written in KnowledgePro. At the start of a DEB session, the user is presented with a sentence fragment in a natural language window and a menu of possible components to choose from (much like Texas Instrument's Natural Access). Once a component is chosen, the natural language window is updated to reflect their selection. Then they are shown a subcomponent menu which is made up of only the subcomponents related to the selected component. When this choice is made, similar things happen and a menu of related actions is shown. In this way the interface guides the user's selection and echoes their menu picks by way of an english sentence. Finally the user is asked to choose a tool and to enter the date of the downtime problem occurrence.

DEB - Diagnostic Experience Browser

The problem is that a router motor is not operating on tool TO1A5103.

Top Ten Browsing

I would like to browse case ...

Similarity Value	Case	Date	Tool	Problem Component / Subcomponent / Actior
0.86	88/197	06/21/88	T01A5002	ROUTER/MOTOR/NOT-OPERATING
0.81	88/226	07/19/88	T03A5012	ROUTER/NOT-OPERATING
0.80	88/196	06/20/88	T01A5002	ROUTER/NOT-OPERATING
0.79	88/288	09/02/88	T01A5002	ROUTER/NOT-OPERATING
0.79	88/319	09/30/88	T01A5002	ROUTER/NOT-OPERATING
0.74	88/091	03/17/88	T01A5103	ROUTER/NOT-MOVING
0.74	88/076	03/03/88	T01A5103	ROUTER/NOT-MOVING
0.62	88/031	02/01/88	T02A5006	ROUTER/NOT-OPERATING
0.69	88/062	02/24/88	T03A5014	SAW/NOT-ENGAGE
0.62	88/066	02/26/88	T03A5014	SAW/NOT-ENGAGE
0.56	88/040	02/08/88	T01A5103	ROUTER/NOT-BACK-GROOVE
0.50	88/095	03/21/88	T01A5103	ROUTER/NOT-CUT-PER-ENG-SPECS
0.42	88/284	08/29/88	T04A5016	ROUTER/MOTOR/SPARKING

Figure 7. Top Ten Most Similar Cases Menu.

When DEB has processed the user's request, a menu appears showing the top ten most similar past cases (Figure 7). When the selection of a case to browse is made, the full textual episode description is retrieved from the data base and shown in a window. Using hypertext, the user can choose to generate another top ten menu based on this trouble-shooting account's problem feature-triplet or cause feature-triplet. If this case references another case (by mentioning its case number), the user can highlight this hypertext thread and DEB will show the corresponding case in the same manner as the case from which it was spawned. This browsing can continue until the user is done.

Issues and Lessons Learned

During the evolution of this task many issues were discussed, tested and evaluated. One of the more prevalent was the blurry line that separated problems and causes. At first thought this does not seem to be an issue, however, it was soon discovered that through the investigation and categorization of problems and causes there was a relationship much like a chain. A chain in that, depending on where a problem was discovered, different but related problem-cause pairs would be named. For example, there is a sensor that detects how far the weld torch is from the metal part. Suppose this sensor fails causing the torch to dive into the part. If the sensor was controlled by a computer and the controlling parameters were input by an engineer, there are more than one problem-cause combination for this anomaly. The choice depends on where in the chain of events the problem is discovered. Often a cause is found that followed from another factor that was at first hidden (perhaps a power surge, in this example).

For these reasons the problem and cause category features were not stored in separate similarity networks, but grouped together in one. This allows the user to follow the chain of events possible for a class of downtime occurrences by allowing the user to browse similar cases which may have been described or categorized differently. The similarity networks perform well in this task.

It was also discovered that a better batch of cases to browse could be attained by using the similarity networks on a large base of cases and assigning thresholds to improve performance. These thresholds may mean limiting the number of similar features each individual problem feature can have to only the most similar. Another way of instituting thresholds is to limit the number of cases that are fully processed and sorted for the top ten. In this way, only those above a certain similarity value after only the symptom-triplets are compared. These types of enhancements yield the best results for a large historical base of events.

Future Enhancements

Several enhancements have been decided upon to make DEB a more intelligent assistant. First, a report module to give statistical analysis of problem/cause trends and a natural language explanation system will be written. Also to be added is a weld team formation module that would be an expert system to decide which members of the weld team are needed to diagnose a problem. This task is currently performed by the weld team coordinator who calls members of the team based on the problem description called in. A weld team formation expert system would alleviate the problem of wasting an expert's time going to a tool site when they are not needed.

It would be useful for the system to perform the case problem and cause categorization automatically. A great amount of expert-derived rule-based knowledge about the domain would be needed along with is-a hierarchies and natural language keyword extraction routines.

Yet another enhancement our group is considering is to combine DEB with another AI system developed here called ELMO. ELMO is a case-based memory building tool and stands for Episodic Long-term Memory Organizer (Pulaski 88). Instead of a similarity network, ELMO uses hierarchies of knowledge to make generalizations about cases. The cooperation of the two knowledge representations would provide added reasoning capability for future applications.

Conclusion

This type of historical browsing capability lends itself to many memory intensive tasks whether they are reasoned off-line or in real-time. For an intelligent assistant, the easy and timely access of experience is essential for tasks in the domain of aerospace manufacturing, as well as other domains. Moreover, this ability is useful for domains in which events are not assembly-line proven but are similar enough that past experience can be used to produce a quality product.

References

(Bailey 88) Bailey, D., Thompson, D., and Feinstein, J. "Similarity Networks." PC AI, July/August, 1988. pp 29-32.

(Pulaski/Casadaban 88) Pulaski, K., and Casadaban, C. "Case-Based Reasoning: The Marriage of Knowledge Base and Data Base," in proceedings of Fourth Conference on Artificial Intelligence for Space Applications, Huntsville, Alabama, November 15-16, 1988, NASA Conference Publication 3013, pp. 183-190.

(Pulaski 88) Pulaski, K., "ELMO: An Episodic Long-Term Memory Organizer for Case-Based Reasoning," AAAI Case-Based Reasoning Workshop, St. Paul, Minnesota, August 23, 1988, pp. 107-112.

(Schmucker 84) Schmucker, K. J., Fuzzy Sets, Natural Language Computations, and Risk Analysis. Computer Science Press, Rockville, Maryland, 1984.