QUANTIFYING AND SIMULATING THE BEHAVIOR OF KNOWLEDGE-BASED INTERPRETATION SYSTEMS*

V.R. Lesser, S. Reed and J. Pavlin

Computer and Information Science Department
University of Massachusetts
Amherst, Mass. 01003

ABSTRACT

The beginnings of a methodology for quantifying the performance of knowledge-sources (KSs) and schedulers in a knowledge-based interpretation system are presented. As part of this methodology, measures for the "reliability" of an intermediate state of system processing and the effectiveness of KSs and schedulers are developed. Based on the measures, techniques for simulating KSs and schedulers of arbitrary effectiveness are described.

I INTRODUCTION

The development and performance-tuning of a knowledge-based interpretation system like the Hearsay-II speech understanding system [1] is still an art. There currently does not exist sufficient formal methodology for relating the performance characteristics of such a system to the performance characteristics of its components; i.e., knowledge-sources (KSs) and schedulers**. For that matter, there does not even exist an adequate framework for quantifying the performance of the system and its components in an uniform and integrated way. Thus, when the initial operational configuration, C1, of the Hearsay-II speech understanding system had poor performance, there existed no methodology for detailing in a quantifiable way what types of performance improvements in specific components would be needed to improve significantly the overall system performance. Therefore, the development of the completely reorganized C2 configuration, which turned out to have much superior performance, was based on "seat of the pants" intuitions. These intuitions were testable only when the new set of KSs were integrated together into a working system.

In the following sections we present the beginnings of a methodology for quantifying the performance of KSs and schedulers. We then show how this methodology can be used to simulate the performance of an upgraded component in a working system, so that more accurate estimates can be made of the overall performance improvement that would be realized if the component were actually upgraded.

II A MODEL FOR A HEARSAY-LIKE KNOWLEDGE-BASED SYSTEM

We will take as the competence goal of an interpretation system the construction of the most credible complete interpretation of the sensory data. The input hypotheses (preprocessed sensory data) are assumed to be errorful and/or incomplete, and are viewed as a collection of competitor sets. A competitor set contains alternative hypotheses describing possible interpretations for mutually-exclusive aspects*** of the sensory data. Associated with each input hypothesis is a belief-value (which ranges from 0 to 1) indicating the system's initial belief in the correctness of the event the hypothesis represents. The initial belief-values are generated by preprocessing the sensory data. This view of input hypotheses is similar to that taken in a relaxation process [6], but subsequent processing in Hearsay-II-like systems is different.

In our model for a knowledge-based system, similar to the Hearsay-II model [1], a complete and consistent interpretation is incrementally constructed by aggregating lower-level hypotheses

---

* This research was sponsored by the National Science Foundation under Grant MCS78-0412 and by the Defense Advanced Research Projects Agency (DOD), monitored by the Office of Naval Research Under Contract NR049-041.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, the Defense Advanced Research Projects Agency, or the US Government.

** Fox [2] has made some efforts in this direction, but we feel that his model of these systems is too abstract to capture adequately many of the important issues. (A detailed discussion can be found in [4].)

*** The specific nature of these aspects depends on the particular interpretation problem to be solved. For example, in the speech understanding task, each acoustic-phonetic segment of the speech signal could be an aspect of the sensory data; the associated competitor sets contain hypotheses that represent possible phonemes.

into more abstract and encompassing higher-level hypotheses (partial interpretations). The lower-level hypotheses are said to support the higher-level hypotheses. This aggregation process, accomplished by synthesis KSs, involves the detection of local consistency (or inconsistency) relationships among hypotheses*. In KS processing, the belief-values of supporting hypotheses are not changed as a result of detection of local consistency, as would be the case in a relaxation process. The construction of a higher-level hypothesis and its associated belief-value is an explicit encoding of the nature and degree of consistency found among its supporting hypotheses. The hope is that this incremental aggregation process resolves the uncertainty among competing interpretations and, simultaneously, distinguishes between correct and incorrect interpretations.

We have not discussed KSs as solely reducing the uncertainty in the system, because uncertainty is a measure of the distribution of belief-values and does not reflect the accuracy of hypotheses. We feel that KS processing causes changes in both certainty and accuracy in a system's database and we have developed a measure, called "reliability", that combines the two. A good KS will produce higher-level hypotheses which are more reliable than the lower-level hypotheses supporting them.

III MEASURING THE SYSTEM STATE

Basic to our view of processing in knowledge-based systems is the concept of system state. The system state at any point in time is the current set of hypotheses and their relationships to the input data. It is through measures of the system state that we can talk in a uniform manner about the performance of KSs and schedulers.

For purposes of measuring reliability, we associate a hidden attribute with each hypothesis which we call its truth-value. This attribute measures the closeness of the hypothesized event to the correct event. For task domains in which a solution is either totally correct or incorrect, we quantify truth-value as either 1 (true) or 0 (false), while in domains in which there are solutions of varying degrees of acceptability, truth-values range between 1 and 0.

One way of evaluating an intermediate state of processing is by measuring the reliability of the set of all possible complete interpretations (final answers) that are supported (at least in part) by hypotheses in the current state. We feel that a direct measure of this sort is not feasible because it is very difficult in general to relate the set of partial interpretations to the very large set of complete interpretations they can support.

We take an alternative approach, called reflecting-back, which is based on two premises. First, the creation of a high-level hypothesis is a result of detecting the constitency among its supporting hypotheses. This creation process is an alternative to actually changing the belief-values of the supporting hypotheses, as occurs in the relaxation paradigm. Thus, the creation of a high-level hypothesis implicitly changes the reliability of its supporting hypotheses. This change can be traced down to the input hypotheses whose reliability is implicitly improved to the extent that they are aggregated into reliable high-level hypotheses. Second, we assume that processing which implicitly improves the reliability of the input hypotheses also improves the reliability of the complete interpretations supported by these hypotheses.

In the reflecting-back approach, we associate with each input hypothesis the highest belief hypothesis it supports. The truth- and belief-values of this highest belief hypothesis are reflected-back to the input hypothesis. The process is illustrated in Figure 1.

It should be stressed that the hypotheses' truth-values and reflected-back values are used only for measuring the system state; they are not available to KSs during processing.

Our measure for the reliability of an intermediate system state is based on a measure of the reliability of input competitor sets computed from the reflected-back belief- and truth-values. Intuitively, a measure of reliability for a competitor set should have the following properties, based on both the belief- and truth-values of its hypotheses:

1. With respect to accuracy, reliability should be high if a true hypothesis has high belief-value or if a false hypothesis has a low belief-value, while reliability should be low if a true hypothesis has low belief-value or a false hypothesis has high belief-value;

2. With respect to uncertainty, reliability should be high if one hypothesis in a competitor set has a high belief-value and the rest have low belief-values, while reliability should be low if all the hypotheses have similar belief-values.

A measure for the reliability, RC(S), of a competitor set, S, that captures and adequately combines both of these properties is:

$$RC(S) = 1 - \underset{h \text{ in } S}{\text{avg}} \; |TV(h)-BV(h)|$$

which is equivalent, in the case of binary truth-values, to the correllation of truth- and belief-values:

$$RC(s) = \underset{h \text{ in } S}{\text{avg}} \; [BV(h)*TV(h) + (1-BV(h))*(1-TV(h))]$$

where BV(h) is the belief-value of hypothesis h and

---

* To simplify this presentation, we focus here on synthesis KSs only, though prediction, verification, and extrapolation KSs also have a place in our model.

TV(h) is the truth-value of hypothesis h. Other measures may also be applicable, but of those we considered this one best captures our intuitive notion of reliability.

Based on this measure of competitor set reliability, we can construct, for instance, a measure of processing effectiveness associated with the current intermediate system state. This measure is the average over the input competitor sets of the difference between the initial reliability and the reliability of the intermediate state. The initial reliability is the average reliability of the competitor sets formed from input hypotheses, using the initial belief- and truth-values. The reliability of the intermediate state is the average reliability of the competitor sets formed from the input hypotheses, where the reflected-back truth- and belief-values of hypotheses are used in place of the original ones. In Figure 1, if the three competitor sets were the only ones on the input level and each one contained just the two hypotheses shown, then the initial reliability, RI, is .517, the reliability of the intermediate state, RS, is .575, and processing effectiveness, PE, is .058:

RI = 1/3[(.7+.6)/2 + (.5+.4)/2 + (.7+.2)/2] = .517

RS = 1/3[(.6+.5)/2 + (.7+.5)/2 + (.45+.7)/2] = .575

PE = RS - RI = .058

A positive PE value, as in this example, indicates that some uncertainty and error has been resolved. The larger the value, the more resolution, the more effective is the processing.

IV MEASURING THE RESOLVING POWER OF A KS

We define the instantaneous resolving power of a KS as a change in reliability due to a single KS execution. This change is measured on competitor sets constructed from the KS input hypotheses. Thus, instead of calculating the reflected-back reliability of the entire system state, the procedure is localized only to the subset of the state directly affected by the KS execution. We

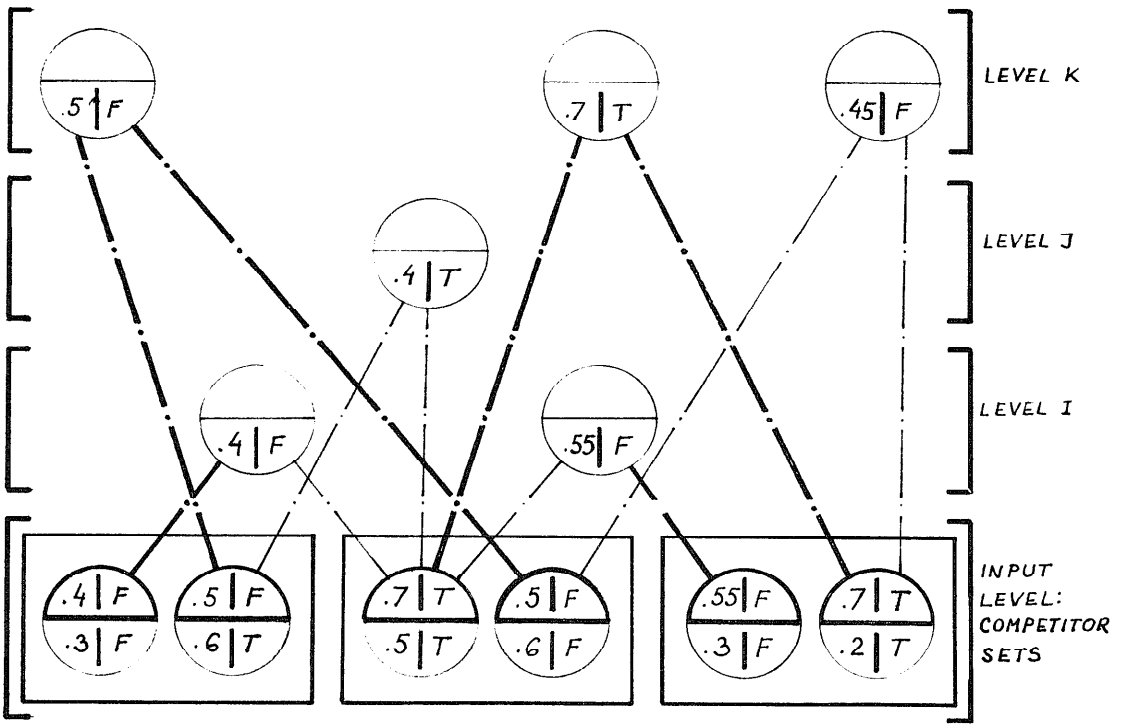

Figure 1: an example of the reflecting back process. The system state contains three competitor sets (represented by boxes) on the input level, and a number of higher level hypotheses. The hypotheses are represented as circles with belief-values on the left and truth-values (T=1 and F=0) on the right. The initial values are in the bottom of the circle, the reflected-back values are in the top half. The thick lines show the support link to the highest belief-value hypothesis supported by each input hypothesis and indicate the source of the reflected-back values. Hypotheses which intervene between the inputs and their highest-belief supported hypotheses are not shown.

measure the change in reliablilty as a result of KS execution by measuring before KS processing the reliability of the KS input competitor sets, then measuring after KS processing the reliability of these sets based on values reflected-back from the KS output hypotheses, and finally taking the difference between the results obtained.

The resolving power of a KS can now be defined as its average instantaneous resolving power over a series of system executions. This view of KS resolving power does not take into account the global impact of KS execution on the entire system state and on future processing. Rather, it is a local, instantaneous measure of the effects of KS execution. The global effects occur through KS interactions, which we believe should be separated from our measure of the resolving power of a single KS.

V SIMULATING A KS

Given a formal measure of KS resolving power, we can simulate KSs of any desired power. This is accomplished by introducing an "oracle" which knows how to judge the closeness of a hypothesized interpretation to the correct interpretation (this is the source of the truth-values). Our reliability measures can thus be calculated during processing rather than in retrospect, after the system has completed processing. Therefore, a system does not have to complete execution in order to be evaluated.

A KS is simulated in two stages: candidate generator and resolver. The candidate generator produces plausible hypotheses for KS output. It can be implemented either by using a real KS whose performance needs to be upgraded or by constructing a simplified KS incoporating relatively simple domain knowledge*. The next stage, the resolver, uses information provided by the oracle to alter the belief-values of the hypotheses output by the candidate generator (with minimal disturbance of their belief-values) to achieve, on the average, a desired KS resolving power. With this alteration process, we can simulate the detection of more sophisticated forms of local consistency than is provided by the candidate generator. This technique allows us to simulate, in a real system, either an upgraded version of a KS (where the old KS is used as candidate generator) or a totally new KS (with partially developed knowledge), in order to understand the system performance implications of a proposed change.

We believe that our approach to simulating KSs of different resolving power, which makes heavy use of an oracle, will prove useful in designing and debugging knowledge-based systems**. However, there are some limitations:

1. Our simulation of KS resolving power is based on a combination of simple knowledge about local consistency and reference to an oracle, while real KSs infer truth from local consistency alone (and falsehood from local inconsistency).

2. The behavior of different simulated KSs sharing similar errors in knowledge will not be correlated due to our statistical approach to KS simulation.

Given these limitations, we do not expect a simulated KS to behave exactly the same as a real KS. We hope, however, the essential behavior of a KS has been captured so that system phenomena are adequately modelled.

In order to validate our models of KS power, we plan to analyze the behavior of KSs in some existing knowledge-based systems. A measures of KS power will be taken for an existing KS and then the KS will be replaced by a simulated KS of the same power, and the overall system behavior compared in the two cases. The results of these experiments should give us some understanding of the extent to which data derived from our simulation studies can be used to predict the behavior of real systems.

VI SIMULATION OF ACCURACY IN THE SCHEDULER

Reliability measures can also be used in the simulation of a scheduler of a specific accuracy. The task of a scheduler is choosing a KS instantiation for execution. A KS instantiation is a KS-stimulus pair, where the stimulus is the set of hypotheses which caused the KS to be considered for scheduling. The scheduler evaluates alternative instantiations according to its knowledge of the characteristics of the KSs, the stimuli, and the current state of processing. The effects of future processing are not factored into this model of scheduling; we take an instantaneous view of scheduling decisions. Because of this, we are unable to model scheduling algorithms such as the "shortfall density scoring method" [7] which use information about future processing. We hope to develop a formulation that includes this type of information.

A good scheduler chooses for execution the KS instantiation that will most improve the reliability of the current system state. The accuracy of a single scheduling decision is defined relative to the performance of an optimum scheduler, which uses accurate information about the resolving power of the KSs and the reliability of the KS stimuli and system state. The accuracy of a scheduler is the average of the accuracy of many scheduling decisions.

We view the optimum scheduling process in two steps:

---

* In many cases, it is relatively easy to design a KS which provides moderate accuracy. Most of the effort in knowledge engineering is spent in increasing this accuracy to gain superior performance.

** The work of Paxton [5] comes the closest to our approach, but was much more limited.

1. For each KS instantiation on the scheduling queue, make accurate predictions concerning its instantaneous resolving power. These predictions involve determining the truth-value of the stimulus hypotheses (using the oracle) and knowledge of the resolving power of the KS.

2. Make accurate predictions as to the global system state which would result from scheduling each instantiation given the predictions of step 1. These predictions will determine optimum ratings for the instantiations and result in an optimum schedule.

Our approach to modelling the scheduler is to obtain statistically accurate ratings for the instantiations, based on the optimum schedule, and then choose for execution an instantiation from within the ordering which results. The position in the ordering of the chosen instantiation depends on the desired accuracy of the scheduler being modelled; the closer to the top of the order, the more accurate the scheduler.

We feel it would be an error to model scheduling only as a function of the truth-value of stimulus hypotheses. Real schedulers do not have access to the truth-values of hypotheses, but only infer truth from belief-values and processing history. The point is that two instantiations of the same KS, whose stimulus hypotheses have equivalent characteristics (same belief-value, level of abstraction, database region, processing history, etc.) except for their truth-values would be rated the same by even the best scheduler. Additionally, in order to determine the rating of a KS instantiation, real schedulers [3] consider other factors, besides the characteristics of the stimulus hypotheses. For example, schedulers take into account such factors as the balance between depth-first vs. breadth-first processing or between executing KSs that work in areas with rich processing history vs. executing KSs that work where little processing has been done. These additional considerations are, in fact, heuristics which attempt to capture the concept of improvement in the reliability of the system state. Thus, in our view, a scheduler should be characterized in terms of its ability to estimate the improvement in system state reliability, rather than its ability to detect the truthfulness of the instantiation's stimulus hypotheses.

We could have modelled the scheduler just as we modelled KSs, with a candidate evaluator and a scheduling resolver. The candidate evaluator would take the generated KS instantiations and give them ratings based on simple scheduling knowledge. The scheduling resolver would minimally alter these ratings (with statistical perturbation) to produce an ordering for the instantiations which corresponds to a desired scheduler accuracy. For several reasons, too complicated to discuss in this short paper, we have not used such an approach for modelling schedulers. Further details of this issue and a more detailed formulation of scheduling measures are discussed in an extended version of this paper [4].

VII SUMMARY

This work represents the beginnings of a methodology for understanding in quantitative terms the relationship between performance of a knowledge-based system and the characteristics of its components. This quantification may also allow us to develop simulations of these systems which can accurately predict the performance of alternative designs.

ACKNOWLEDGMENTS

We would like to recognize the helpful comments on various drafts of this paper given by Daniel Corkill and Lee Erman.

REFERENCES

[1] Erman, L. D., F. Hayes-Roth, V. R. Lesser and R. Reddy (1980), "The Hearsay-II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty," Computing Surveys, 12:2, June 1980.

[2] Fox, M. S. (1979), "Organizational Structuring: Designing Large Complex Software," Technical Report CMU-CS-79-155, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

[3] Hayes-Roth, F. and V. R. Lesser (1977), "Focus of Attention in the Hearsay-II Speech Understanding System," In Proceedings of the Fifth International Joint Conference on Artificial Intelligence-1977, p. 27-35, Cambridge, Massachusetts, 1977.

[4] Lesser, V. R., J. Pavlin and S. Reed (1980), "First Steps Towards Quantifying the Behavior of Knowledge-Based Systems," Technical Report, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts.

[5] Paxton, W. H. (1978), "The Executive System," in D. E. Walker, (editor), Understanding Spoken Language, Elsevier, North-Holland, N. Y., 1978.

[6] Rosenfeld, A. R., R. A. Hummel, and S. W. Zucker (1976), "Scene Labeling by Relaxation Operators," IEEE Transactions on Systems, Man and Cybernetics, SMC-6, pg. 420-433, 1976.

[7] Woods, W.A. (1977), "Shortfall and Density Scoring Strategies for Speech Understanding Control," In Proceedings of the Fifth International Joint Conference on Artificial Intelligence-1977, p. 18-26, Cambridge, Massachusetts, 1977.