GENERATION OF CAUSAL MODEL FOR SECOND GENERATION EXPERT SYSTEMS

H. Bilić, I. Boljat, Z. Grbac, A. Jadrić,
D. Marković, P. Slapničar

Fakultet elektrotehnike, strojarstva i brodogradnje
58000 Split, Rudera Boškovića bb

Second generation expert systems are able to combine reasoning based on heuristic rules with reasoning based on system models. The paper outlines a graph form of a causal model of a technical system. The problem solving algorithms as well as deep reasoning based on the model are outlined.

Keywords /deep reasoning/heuristic rules/top-bottom examination/

1. INTRODUCTION

Expert systems are software systems consisting of two basic components: the knowledge base consisting of facts and rules as well as the shell processing the knowledge base. It differs from conventional programs with regard to organization, performance and user-oriented dialogue (Hayes-Roth, 1984, Lavrač et al., 1986).

First generation expert systems rely completely on heuristic knowledge given in the form of rules, most often in IF-THEN form (Bilić et al., 1989). However, second generation expert systems, in addition to heuristic rules introduce the causal model of a system, covering the whole research domain.

It is a drawback of the first generation expert systems that they do not function beyond the domain defined by heuristic rules (Lavrač et al., 1986), while second generation expert systems, if they do not find a solution within the rule defined area, do not stop working (Steels, 1985), but pass on to reasoning via the system model (deep reasoning) (Steels et al., 1986), which results in gradual degradation of expert system performance.

With second generation expert systems the manner of problem solution is more logical and easier to understand, since they can use the system model which has the possibility of deeper understanding of the system behaviour.

The most important advantage of the second generation expert systems lies, however, in knowledge acquisition. With first generation expert systems, finding new heuristic rules is an extremely difficult task: it takes a long time to come up with solid rules; the existing ruleset is never complete and it often shows inconsistencies. These inconsistencies are due to different experiences of experts creating the rules. Second generation expert systems, on the other hand, solve the problem using the model, at the same time having a learning capacity, i.e. a capacity for finding new heuristic rules extracted from reasoning based on the system model.

2. CAUSAL MODEL

2.1.Causal Model Representation

Functioning of complex technical systems can be represented by a causal model (Steels et al., 1986) consisting of a set of interdependent properties of the system components. The value of a property is determined by the values of one or more other properties. Causal relations showing the properties of the components can be represented with a network of nodes (Neapolitan, 1986).

Fig. 1 shows a part of a TV set causal model (supply circuit). The state of the device as well as expected values of particular properties are given.

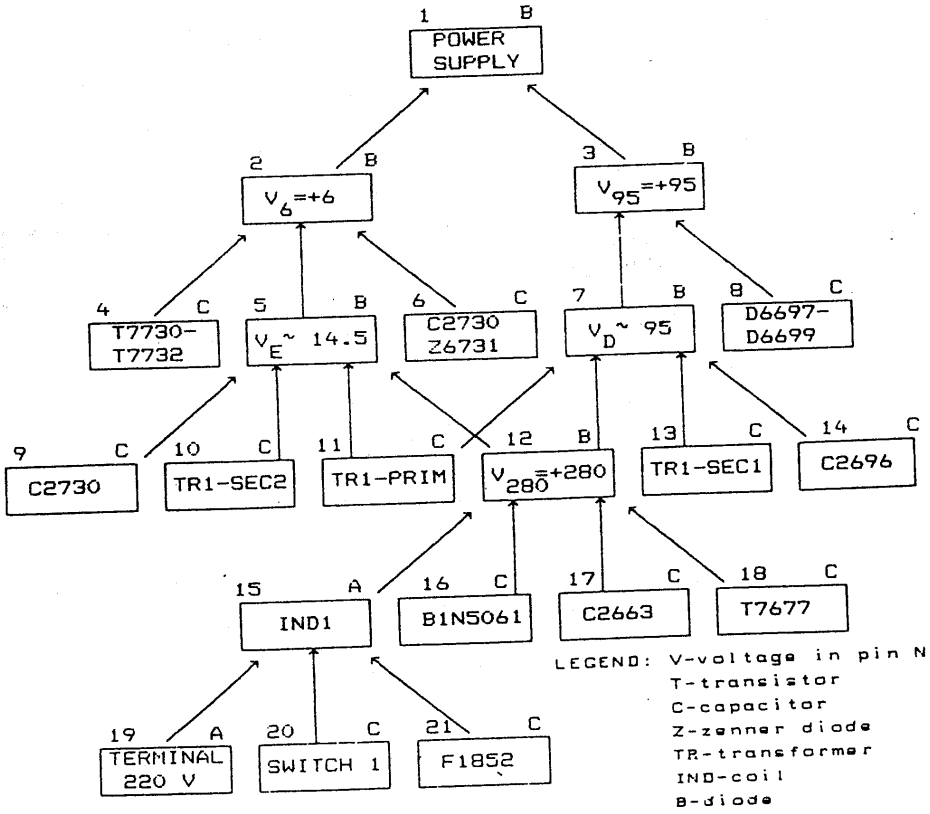


Fig. 1 A part of a TV set causal model

Properties of the model are shown in nodes from 1 to 21.
Designations: A → easily observable properties
B → less easily observable properties

(e. g. in voltage measurements)
C → barely observable properties
(e. g. in component checking)

2.2. DEEP REASONING

Deep reasoning can be carried out using the following principles:

- if the effect of a property (component) is normal, then there is reason to believe that this property (component) is normal;
- if the effect of a property is anomalous, then the property itself may be anomalous too.

A deep reasoning algorithm begins with an anomalous property and investigates the causes of this property until observable properties are arrived at. If these properties are anomalous, then the cause of the anomaly has been found. If these properties are normal (i.e. in accordance with the expected values), the solution of the problem lies elsewhere, i.e. in internal components which are not observable. The number of suspect components can be reduced if there are other observable components caused by the same property. Suppose for example, the property of a node (1) is anomalous and needs to be explained. Furthermore, we have established that the property of the node (3) is normal. When checking the nodes it is not necessary to prove that nodes (11) and (12) are normal, since their properties are at the same time the causes of node (7) or (3), while we have established that the property of node (3) is normal. In this way the number of the suspect components has been reduced to nodes (4), (6), (9) and (10).

3. CAUSAL MODEL REPRESENTATION AND DEEP REASONING ALGORITHMS

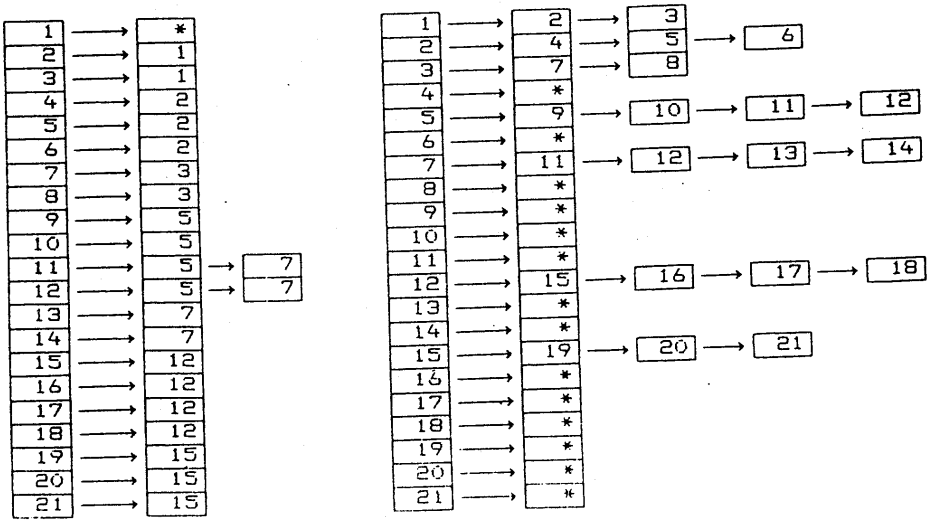


Fig. 2 Bottom-up and top-down representation of a directed graph structure

The causal model shown in Fig. 1 can be represented by a directed graph (Fig. 2). In order to efficiently meet the requirements of the top-down graph (the cause of malfunction is being investigated) and bottom-up graph (the manifestation of the malfunction is being looked for), a twofold representation of the same graph is given (one graph can computationally be reconstructed from another). The node record is permanently set by the fields: the node number and the property description, which are, during the examination procedure, added a temporary value ('T', 'F', 'N', 'E', ' ').

If we wish to predict all the forms of a known anomaly we shall use a bottom-up algorithm based on making the QI queue consisting of anomalous nodes. Initially the anomaly (the number of the node with improper property) is written into the QI queue. Then to each node in QI are added all the successors missing from the QI queue.

```
PROCEDURE Bottom-Up (fault);     (* retrieval for causal model *)
Init (Q_I);        (*queue of number of the node with improper *)
                                                   (*properties *)
Add (Q_I, fault),            (*improper property's node number  *)
a1:= First (Q_I);
WHILE NOT Empty (Q_I) DO
     a2:= First(a1's successor);     (* from Bottom-Up graph  *)
     WHILE more a1's successors DO
          IF NOT Containned (Q_I, a2) THEN
               Add (Q_I, a2)
          ENDIF
          a2:= Next (a1's successor)
     ENDWHILE;
     WRITE (a1);          (* This is one successor - conseqence *)
     a1:= Next (Q_I)
ENDWHILE;
ENDPROCEDURE;                      (* end of Bottom-Up  algorithm  *)
```

So, in case of an improper property the QI queue in node 15 would be:

15 12 5 7 2 3 1.

If we wish to find out the cause of an improper property (component), the top-down algorithm, based on QI queue will be used. The numbers of improper nodes, as well as numbers of nodes with unknown properties having no normal successors, are written in this queue. The cause of the anomaly (solution) is that member of QI queue which has no improper predecessor.

```
PROCEDURE Top_Down (symptom);   (* retrieval for causal model *)
                  (*symptom= initial node's number with a fault *)
Init Q_P;     (* Q_P= queue of potential faulty nodes' numbers *)
FOR all nodes set node.value := ' ';
Add (Q_P, symptom);
change_QP := true;
a:= First (Q_P);
WHILE change_QP AND more Q_P's member DO
     change _QP := false;
     a1 := First (a's predecessor);     (* from Top_Down graph *)
     IF a1.value =' ' THEN
          INPUT (a1.value)  (* 'T'-true, 'F'-false,'N'- unknown,
```

```
        'E'-eliminated = false, but with a faulty predecessor *)
    ENDIF;
    WHILE more a's predecessor DO
        IF a1.value = 'N' THEN
            IF NOT Correct_Successor (a1) THEN
                Add (Q_P, a1);
                change_QP := true;
            ELSE
                a1.value := 'T'  (* has correct successor *)
            ENDIF
        ELSE
            IF a1.value = 'F' THEN
                Add (Q_P, a1);
                change_QP := true
            ENDIF;
            IF a1.value = 'E' THEN change_QP := true ENDIF
        ENDIF;
        a1 := Next (a's predecessor);
    ENDWHILE;
    IF change_QP THEN
        a.value := 'E';
        a := Next (Q_P)
    ENDIF
ENDWHILE;
WRITE (a);                          (* This is the cause of the fault *)
ENDPROCEDURE;                                                (* Top_Down *)

FUNCTION Correct_Successor ( x: node): boolean;
                    (* true if any x's successor is correct *)
Init (Q_U);                 (* queue of unknown nodes' numbers *)
Add (Q_U, x.number);
Correct_Successor := false;
x1 := First (Q_U);
WHILE NOT Empty (Q_U) AND  NOT Correct_Successor DO
    x2 := first (x1's successor);     (* from Bottom_Up graph *)
    WHILE more x1's successors AND NOT Correct_successor DO
        IF x2.value = 'T' THEN
            Correct_successor := true
        ELSE
            IF x2.value = ' ' THEN
                INPUT (x2.value);                (* 'F'/'T'/'N' *)
            ENDIF;
            IF x2.value = 'T' THEN
                Correct_Successor := true
            ELSE
                IF NOT Contained(Q_U, x2.number)
                    Add (Q_U, x2.number)
                ENDIF
            ENDIF
        ENDIF;
        x2 := Next (x1's successor)
    ENDWHILE;
    x1 := Next (Q_U)
ENDWHILE
ENDFUNCTION;  (* Correct_Successor *)
```

4. CONCLUSION

The paper describes a part of a causal model of a technical system (TV set). The node network method is used for model representation. The way of finding a solution in the domain not covered with heuristic rules is described. Algorithms for TOP-DOWN and BOTTOM-UP examination based on deep reasoning are given. The computer program described can be used as a shell of the causal model applicable in other domains.

It is characteristic of engineering systems that all the nodes are not equally accessible. Therefore an expert system should aim at solving the problem using only easily accessible nodes, and only in case of failure pass to less easily accessible ones. Such an approach will be implemented and described in the paper to follow.

REFERENCES

[1]. Bilić, H., Boljat, I., Grbac, Z., Jadrić, A., Marković, D., Slapničar, P., Saša, S. (1989), "An experiment with knowledge base", Zbornik radova X međunarodnog simpozija Kompjutor na sveučilištu, Cavtat, 1989.

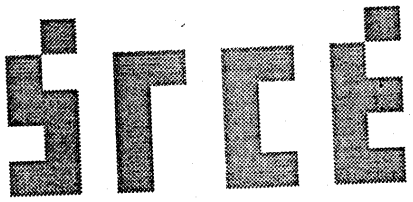
[2]. Hayes-Roth, F. (1984), "The Knowledge-Based Expert Systems", IEEE Computer, vol. 17, No. 9, Sept. 1984, pp. 11-28.

[3]. Lavrač, N, Varšek, A, Gams, M, Kononenko, I, Bratko, I, (1986), "Automatic Construction of the Knowledge Base for a Steel Classification Expert System", The 6-th Int. Workshop on Expert Systems and Their Applications, Avignon, April 1986.

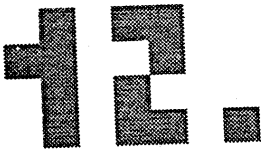
[4]. Neapolitan,.R. A. (1986), "Forward-chaining versus graph approach as the inference engine in expert systems", Proc. Applications of Artificial Intelligence III, SPIE vol. 635, pp. 62-69, apr. 1986.

[5]. Steels, L. (1985), "Second Generation Expert Systems", Future Generation Computer Systems, vol.1, no.4, pp. 213-221, 1985.

[6]. Steels, L. and W. Van De Velde (1986), "Learning In Second Generation Expert Systems", chapter 10 In Kowalik, J. S. (Editor) "Knowledge Based Problems Solving", Prentice Hall Inc, Englewood Cliffs, N.J, 1986.



SVEUČILIŠNI RAČUNSKI CENTAR
UNIVERSITY COMPUTING CENTRE



MEĐUNARODNI SIMPOZIJ KOMPJUTER NA SVEUČILIŠTU

INTERNATIONAL SYMPOSIUM COMPUTER AT THE UNIVERSITY

CAVTAT 1990.

Under the auspices of:

ETAN

YUGOSLAV ACADEMY OF SCIENCE AND ARTS

Program Committee

Allen, L.E.	USA
Aurer, B.	Yugoslavia
Baručija, M.	Yugoslavia
Bratko, I.	Yugoslavia
Budin, L.	Yugoslavia
Čerić, V.	Yugoslavia
De Wilde, W.P.	Belgium
Dujmović, J.	Yugoslavia
Đorđević, B.	Yugoslavia
Ferligoj, A.	Yugoslavia
Haznadar, Ž.	Yugoslavia
Javor, A.	Hungary
Krapac, D.	Yugoslavia
Lauro, N.	Italy
Lužar, V.	Yugoslavia
Momirović, K.	Yugoslavia
Paul, R.	United Kingdom
Petrić, J.	Yugoslavia
Seljak, T.	Yugoslavia
Srića, V.	Yugoslavia
Stahl, H.	GDR
Stipanović, R.	Yugoslavia
Žepić, A.	Yugoslavia
Žugaj, M.	Yugoslavia

Organizing Committee

Aurer, B.
Dobrić, V.
Gaćeša, M.
Lužar, V.
Radić, B.
Vidlanović, D.