PERIODICA POLYTECHNICA SER. CIVIL ENG. VOL. 37, NO. 1, PP. 13–29 (1993)

EXPERIENCES IN EXPERT SYSTEM DEVELOPMENT

Dezső HOLNAPY* and Gábor RÉDEY**

*Technical University of Budapest
Faculty of Civil Engineering
Laboratory of Informatics
H–1111 Budapest, Műegyetem rakpart 3. K.mfsz. 26.
Tel.: (361) 166 5011/1933

** ABB Láng Ltd., Budapest
H–1124 Budapest, Kiss János alt. u. 55., Hungary
Tel: (361) 175 9676

Received: December 4, 1992
Revised: February 1, 1993

Abstract

Experiences of AI technology application in engineering domain are discussed and illustrated by several demonstrative examples. Economic aspects of development are also considered.

Keywords: : AI technology application, engineering knowledge formalization, knowledge base generation, economical aspects.

1. Introduction

Expert system technology can help to solve problems of a given domain based on the knowledge of a (several) domain expert(s) and an inference engine which is able to follow even the expert's way of thinking.

The knowledge base involves domain knowledge elements that can be stored up in many ways. Most of the systems in engineering domain are rule based ones, i.e. knowledge is represented by 'if... then...' type of rules.

The function of the inference engine is to revive the formalized knowledge elements. It is obvious to use one of the well-known logic programming languages as implementation language, but for getting more efficient systems often numerical languages are preferred. This latter may restrict a bit the form of the questions to put, but the promptness of the answers can compensatethe user abundantly for this disadvantage.

The most popular expert system tools are the shells (considered to be domain independent knowledge processing devices) that furnished by domain knowledge and specific inference engine information result in an expert system of the given domain. This implies the usual attitude of outsiders: if one has a suitable shell then 'only' some knowledge representation work is left to the expert. Anybody who has ever tried this knows that here is the bottleneck today, i.e. we are not able to formalize our expert knowledge in rules without any methodology which needs applying special organization methods. This short article gives a summary of our observations in expert knowledge representation.

2. Modelling

One of the most important problems in applied research is the selection of a suitable model — even at the expense of compromises — according to solvability aspects. *Fig. 1* illustrates what we mean by this statement. It compares pure- and applied research with respect to their working methods. *Fig. 1a* shows a modelling strategy of pure research: it takes only the problem itself into account and declares itself to be independent of later possible solution methods. The branching outs represent problems generated by previous ones. The typical method of applied research is represented by *Fig. 1b*. The converging lines refer to an expected fruitful ending that is necessary within reasonable time. Expert system development can essentially be considered as applied research, and its effectivity depends on the variety of expert system tools (artificial intelligence models, shells, etc.) developed in pure research.

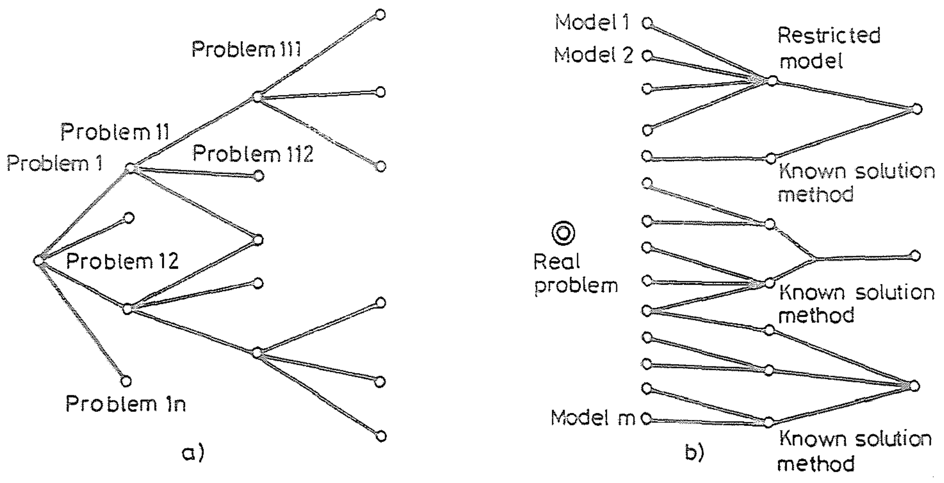


Fig. 1. Scheme of pure and applied research

The domain expert plays a key role during development, both in problem modelling and evaluating results as well. In the days when numerical problem solving was autocratic we used numerical languages for computer aided problem solving, and then also algorithm construction was essential in our work. The reason is that the expert knowledge of the given problem was not able to be formalized without any structural change, i.e. the expert was usually unable to present his knowledge in a form suitable for formalization process. Only a part of his knowledge was well formalized, the other (unformalized) part was his meta knowledge, that is necessary to the application of the former one. *Fig. 2* illustrates how large the formalized part of the knowledge is in a certain engineering problem, namely in case of solitary basement design (a), compared to the knowledge about the application of the former one (b). In textbooks there are only a few lines of algebraic formulae related to solitary basement design, that is some formulae of controlling the related calculations. On the other hand, the algorithm of the whole design process — the engineer's knowledge about the method of the design — is demonstrated by the flow chart. This example shows that constructing an algorithm is not the only way of problem solving. The interaction of a formalized problem solver and the natural intelligence usually seems to be a more efficient device than algorithm constructing.

By now it is the reality that practical problem solving methods can be completed by methods of artificial intelligence. Artificial intelligence methods make possible to solve intellectual problems that do not have any predetermined solution method. Although artificial intelligence systems are strongly attached to formal logic that has been known for 2500 years, its methods went through a tremendous development during the recent decades, SHIRAI (1987). If a problem can be formalized in first-order predicate logic, then only the premises have to be taken into account, and from the collection of inference rules the modus ponens is sufficient to be used so that the problem could easily be solved. An analogous statement holds in case of theorem proving, where the typical question is: whether a certain sentence of a Chomskyan grammar (algebraic formula) is correctly formed. In this latter case the inference rules of the system are the production rules of the Chomskyan grammar. *Fig. 3a* shows a simple formal logical inference, while *Fig. 3b* is a micro-PROLOG program which decides if a certain sentence representing a ceiling constructed by beams and linings is correct or not. Solution methods based solely on certain facts belong to this problem category.

If a problem is more complicated, that means a large number of statements representing knowledge, then this set of statements will be inconsistent and incomplete without using any testing method. We have the same difficulties with logical modelling that we had with numerical problem solving 25 years ago. We have to invent the methods which applied suitably result in a consistent and complete knowledge base of a formal system.

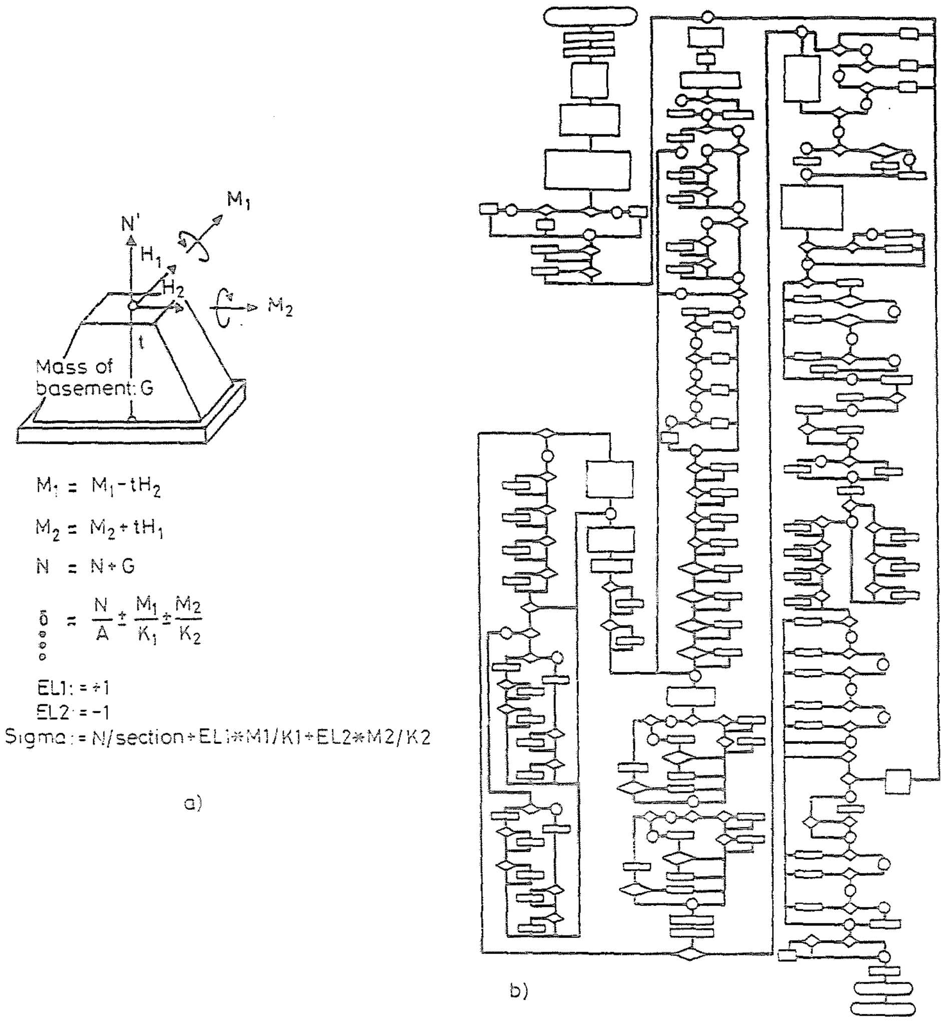


Fig. 2. Solitary basement design: the formulae of mechanical calculations (a) the formalized knowledge and the algorithm of the design process (b)

	((NTRM (ceiling)))
	((NTRM (beam)))
	((NTRM (lining)))
	((NTRM (inner part)))
	((NTRM (field 1)))
	((NTRM (field 2)))
	((→(ceiling)(beam inner-part beam)))
	((→(inner part)(field 1)))
	((→(inner part)(field 2)))
	((→(field 1)(lining)))
	((→(field 1)(field 1 beam lining)))
+ the pavement is wet – (if) it rains	((→(field 2)(lining))
+ it rains	((→(field 2)(field 2 beam lining))
+ the pavement is wet	((→(lining)(EB60/19)))
	((→(lining)(EB30/19)))
	((→(beam)(E7-24)))
(a)	(b)

Fig. 3. Formal deduction by the rule modus ponens

3. How Can a Consistent Knowledge Base be Created?

Problems of the engineering domain can be solved by logical problem solving methods provided that an adequate model of the system can be formalized. Nevertheless, those problems cannot be easily recognized, the associated knowledge of which is easily representable by a well structured set of rules. We show a knowledge base, as a demonstrative example, on civil engineering geology which is to determine names of eruptive rocks. These names depend on the mineral constituents and the structure of the rock. *Fig. 4* shows the Rosenbusch–Schafarzik table that obviously reflects consistent and complete knowledge not involving any uncertainty.

This rock identifier system was written in MPROLOG (SzKI), ES Projects (1986). The name and the structure of the rock is combined in the STRUCTURE relation, while the same name and the list of its mineral constituents are combined in the MINERALS relation, which pair of relations create the 'world'. If the STRUCTURE and the MINERALS relations hold for a given rock, then its name, its structure and the list

Mineral	K-feldspar		Na-Ca-feldspar			-
Rock	quartz	-	quartz	-	olivin	
effusive	riolite	trachite	dacite	andesite	basalt	limburgite
druse	aplites, pegmatites				-	
abyssal	granite	sienite	grano-diorite	diorite	gabbro	peridotite wehrlite piroxenite

Fig. 4. Table of eruptive stones (a simplified version of the Rosenbusch–Schafarzik table)

of its mineral constituents are interrelated. *Fig. 5* shows the MPROLOG program of the eruptive rock identifier.

As a consequence of our experiences we can state that the knowledge that can be summarized in a table can also be expressed in rules. We prepared several other logical systems like the rock identifier so far that can also be considered as knowledge bases, for example: fire protection distance determination, rivet/screw axes arrangement, classifying concrete by testing pieces made of fresh concrete mixture. We have to stress here, that programming is just a routine work in these examples, but transforming unstructured expert knowledge into a well structured representational form is worth mentioning.

Expert systems usually deal with uncertain knowledge. The problems discussed previously do not fall under this category, so we might as well construct algorithms to solve them. Nevertheless, forming intelligent systems in these cases can be explained by the fact that any kind of question can be put to the relations comprising the knowledge base.

4. Systems Dealing with Uncertain Knowledge

Among rule based systems dealing with uncertain knowledge there are numerous expert systems performing diagnostic tasks (MYCIN, LITO, BONNET (1982)). Another representative of these systems is the one developed by the Hospital of Szekszárd (Hungary). We also developed a system like this for geotechnical reasoning of building damages. The knowledge covered by this system can be expressed in a table-like form. For every statement describing the state of the building and its environment there are attached additional information connecting the possible reason of the noticed damage type with the statement in question classified in one of the

```
rock (X,Y,Z):-
     structure (X,Z), mineral (X,Y).
rock (X,Y,Z):-
          structure (X,Z), mineral (X,Y), part (Y,U).
structure (granite, crystalline).
structure (sienite, crystalline).
structure (grano-diorite, crystalline).
structure (diorite, crystalline).
structure (gabbro, crystalline).
structure (peridotite, crystalline).
structure (wehrlite, crystalline).
structure (piroxenite, crystalline).
structure (aplites, porphyritic-crystalline).
structure (pegmatites, porphyritic-crystalline).
structure (riolite, porphyritic).
structure (trachite, porphyritic).
structure (dacite, porphyritic).
structure (andesite, porphyritic).
structure (basalt,porphyritic).
structure (limburgite, porphyritic).
minerals (granite, [quartz, K-feldspar]).
minerals (sienite, K-feldspar]).
minerals (grano-diorite, [quartz, Na–Ca-feldspar]).
minerals (diorite, [Na–Ca-feldspar]).
minerals (gabbro, [Na–Ca-feldspar, olivin]).
minerals (peridotite, [olivin]).
minerals (wehrlite, [olivin]).
minerals (piroxenite, [olivin]).
minerals (aplites, []).
minerals (pegmatites,[]).
minerals (riolite, [quartz, K-feldspar]).
minerals (trachite, [K-feldspar]).
minerals (dacite, [quartz, Na–Ca-feldspar]).
minerals (andesite, [Na–Ca-feldspar]).
minerals (basalt, [Na–Ca-feldspar, olivin]).
minerals (limburgite, [olivin]).
element (X,[X|Y]).
element (X,[X|Y]:-
     element (X,Z).
part ([X|Y],Z):-
     element (X,Z), part (Y,Z).
part ([],X).
```

Fig. 5. Program test of the stone identifier

following categories: 'it can be a reason of the damage', 'it cannot be excluded as a reason of the damage', 'it cannot be the reason of the damage', etc.

A well structured expert knowledge was available to build up this knowledge base, RÉDEY (1989). The geotechnical knowledge fragment that does not involve any uncertainty could easily be represented by an MPROLOG program driven by the MPROLOG SHELL (SzKI). Some parts of this consistent and complete knowledge fragment can be seen in *Fig. 6*, RÉDEY (1989). Another fragment of the same expert knowledge, making use of the facilities supporting inexact reasoning, was also represented in a rule based expert system shell GENESYS (SzÁMALK), GENESYS (1987), (also prepared to use algorithmic routines written in C). According to our experiences, the monomaniac use of logic programming languages is not always worth the trouble in expert system implementations, that is taking the present computer capacities into account, systems efficient enough and capable to give answers saying more than trivialities cannot be implemented in logic programming (PROLOG) or applicative/functional (LISP) languages.

5. An Expert System Approach to the Hungarian Building Regulations

An expert system was developed making use of the experiences outlined above covering a problem arising in the building industry, namely automatic information retrieval from the Hungarian Building Code. The greatest problem we had to face was the organization of research and development.

The expertise of this domain means some kind of decision making, whether a certain object satisfies the prescriptions of the Building Code OÉSZ (1986) according to information obtainable from cadastral maps, plans of building sites, architectural plans, etc. These conclusions are made by the officials in charge of building at local authorities on the basis of the Building Code and several other orders of the Government. It is obvious that an expert system like this can not be used by totally uninitiated people: the user is supposed to be perfectly aware of the current concepts of the system and the competence of its knowledge. Consequently, this expert system cannot be considered as any substitute of the officials involved in this work, that pours permissions automatically. It can be, however, an aid that forces the user working at a local council and having some technical qualification to answer systematically the questions the system puts, and that points out the unsatisfied requirements every time untiringly. The

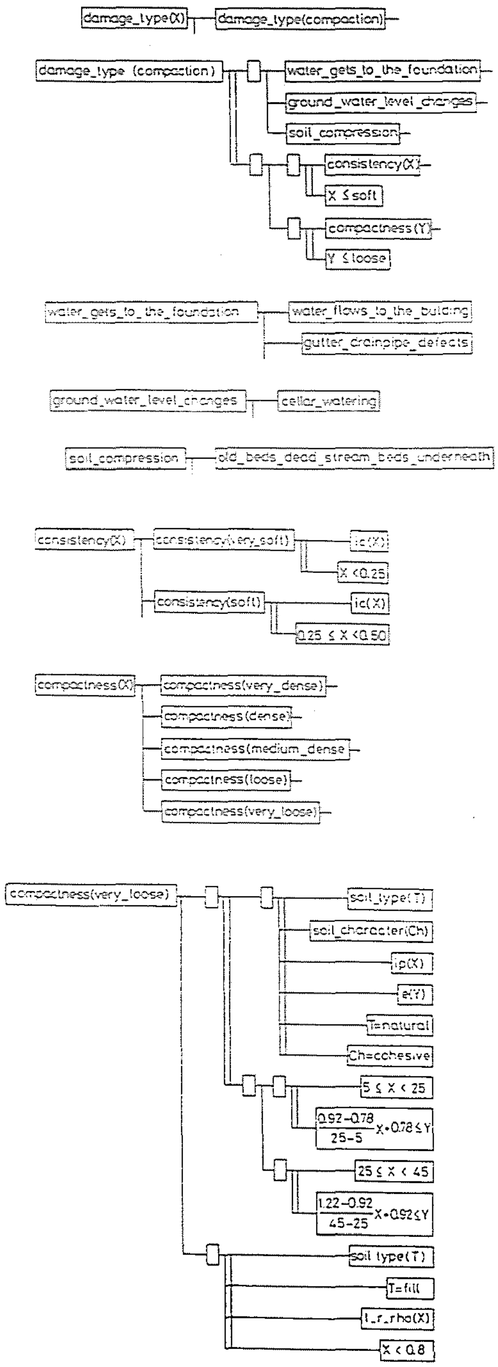


Fig. 6. Derivation tree parts of the ES for building damages

revision of the results is still left to the intelligence and expertness of the official. Hence, this expert system of the Building Code plays the same role in supporting the building authorities in decision making as CAD plays in design.

There is another system in Hungary (JIR, a legal information system), KOVÁCS (1985), SEPRŐDI (1985a-c), VÁRKONYI (1985), supporting legal decision making, similar to the expert system of the Building Code. It is implemented in an imperative language. It also has a micro version, certain parts of which are written in TURBO PROLOG. According to organizational experiences, DESCHAMPES (1987), the following stages can be distinguished if an expert system is to be developed using a lengthy textual knowledge base structured into decimally labelled chapters, subchapters, paragraphs and subparagraphs: text searching system, decision graph representation of the knowledge, rule based system using (possibly) a special inference engine. The following sections discuss these stages in detail.

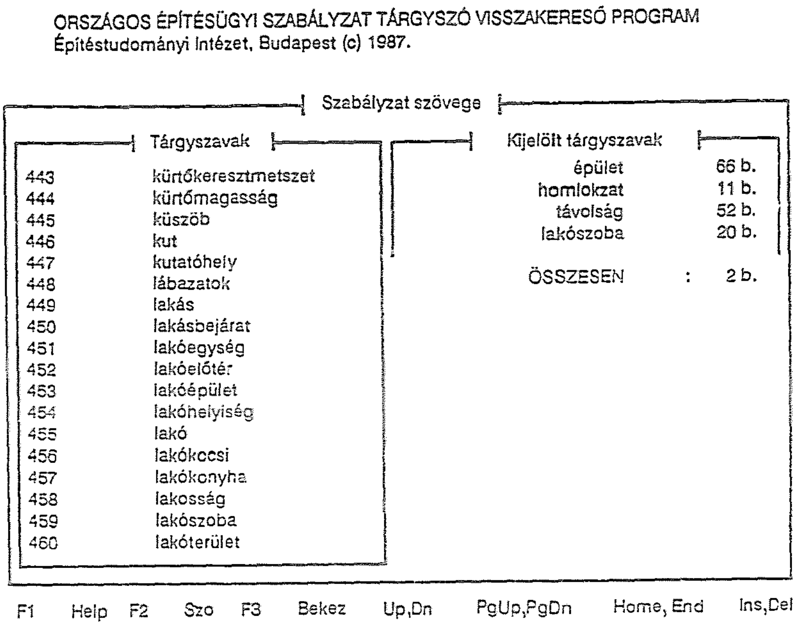
5.1. Text Searching System

The system performs something more than a fully interactive index (i.e. concordance) of the words (synonyms are contracted and non-relevant words deleted, etc.) occurring in the text. Concepts (descriptions) selected from the index and also their combinations with usual logical connectives (and, or, etc.) can be searched, and the associated paragraphs, where they were found (which are considered to be the least text units) are displayed (*Fig. 7*). A system like this can be an efficient aid to people who are to transform the content of a text into a decision graph or rules of 'if... then...' type.

The text searching system is also involved in the expert system as a help function, because the user may need every now and then the exact text of the Code during session. Interviews with local authority officials convinced us about the necessity ty of this kind of systems. This one was tested by the local councils of Bácsalmás and Szentes.

5.2. The System Covering the Authorization of Civil Building Activities Modelled by Decision Graphs

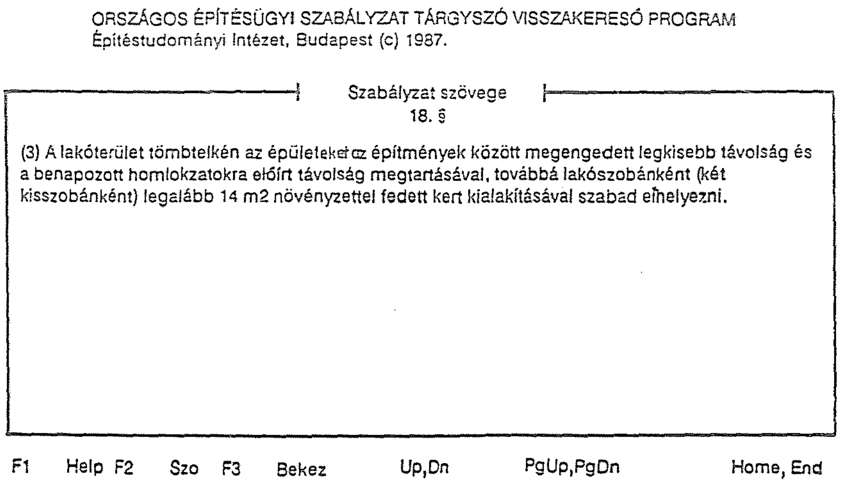
Alphanumeric data input is commonly needed during a computer session. For practical reasons questions concerning these data inputs are put in a predetermined sequence. This kind of data collecting system (question net) is implemented in the shell GENESYS, ASZALÓS (1986), GENESYS (1987). A similar question net, a directed acyclic graph, is represented in



```
ORSZÁGOS ÉPÍTÉSÜGYI SZABÁLYZAT TÁRGYSZÓ VISSZAKERESŐ PROGRAM
Építéstudományi Intézet, Budapest (c) 1987.

                         Szabályzat szövege
          Tárgyszavak                          Kijelölt tárgyszavak
443       kürtőkeresztmetszet                  épület          66 b.
444       kürtőmagasság                        homlokzat       11 b.
445       küszöb                               távolság        52 b.
446       kut                                  lakószoba       20 b.
447       kutatóhely
448       lábazatok                            ÖSSZESEN    :    2 b.
449       lakás
450       lakásbejárat
451       lakóegység
452       lakóelőtér
453       lakóépület
454       lakóhelyiség
455       lakó
456       lakókocsi
457       lakókonyha
458       lakosság
459       lakószoba
460       lakóterület

F1  Help  F2  Szo  F3  Bekez  Up,Dn  PgUp,PgDn  Home, End  Ins,Del
```



```
ORSZÁGOS ÉPÍTÉSÜGYI SZABÁLYZAT TÁRGYSZÓ VISSZAKERESŐ PROGRAM
Építéstudományi Intézet, Budapest (c) 1987.

                         Szabályzat szövege
                               18. §

(3) A lakóterület tömbtelkén az épületekets az építmények között megengedett legkisebb távolság és
a benapozott homlokzatokra előírt távolság megtartásával, továbbá lakószobánként (két
kisszobánként) legalább 14 m2 növényzettel fedett kert kialakításával szabad elhelyezni.

F1  Help  F2  Szo  F3  Bekez  Up,Dn  PgUp,PgDn  Home, End
```

Fig. 7. Displays of the text searching system

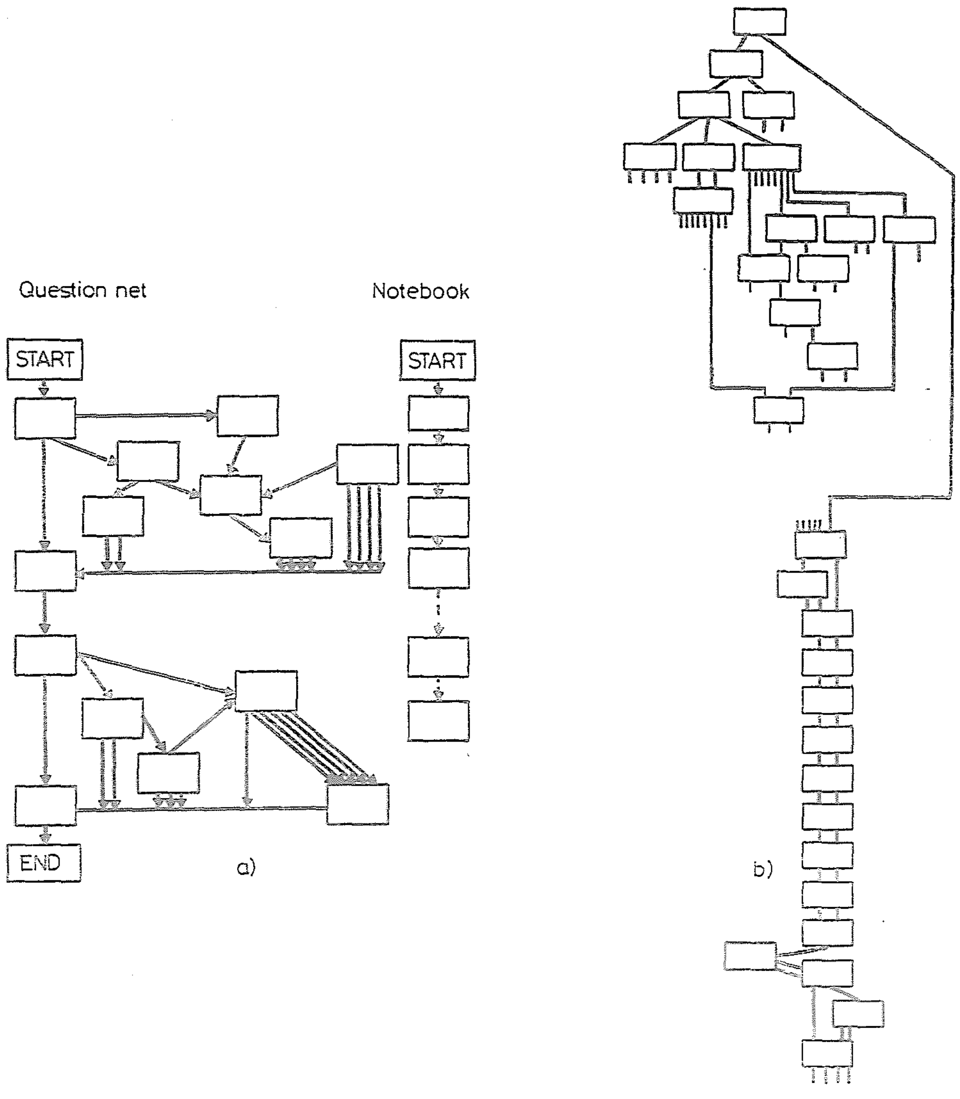


Fig. 8. Decision graph (question net) scheme and its realization

the system processing the knowledge of the Building Code modelled by decision graphs (*Fig. 8*).

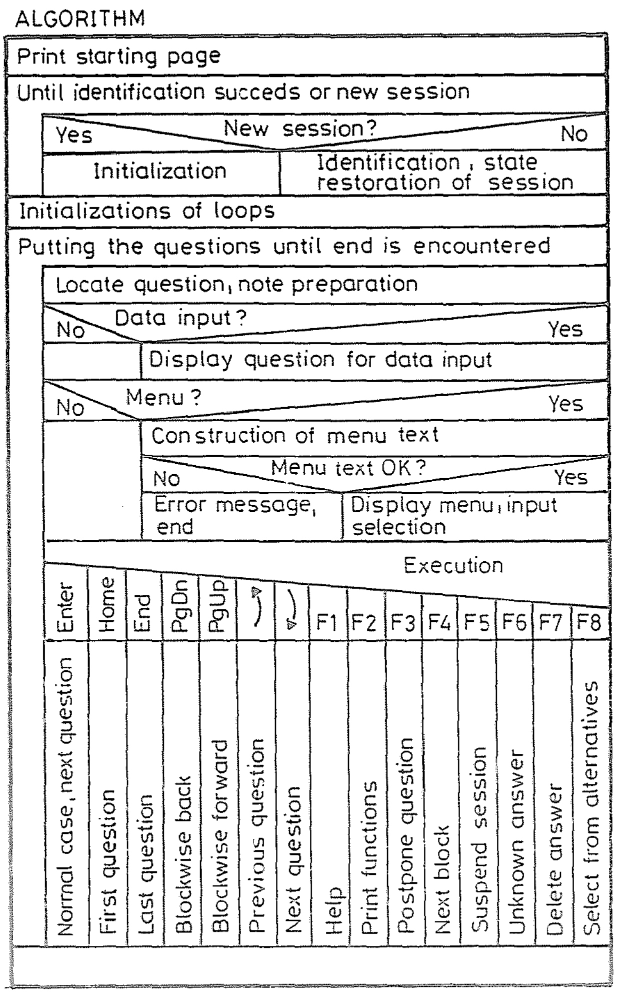
Knowledge representation in decision graphs yields a well structured data collecting system, and it also helps to construct a consistent and complete set of rules for the knowledge base of an intelligent system. If the knowledge does not contain any uncertainty, as in our case, then there is no point in developing an artificial intelligence system, because it is unnecessary and destroys efficiency at present average computation speeds. Our system, that is by now on the market, is written in PASCAL.

The knowledge extractable from written information (of standards, textbooks, literary works) is always incomplete. These texts are both without certain evident information, and without the knowledge (meta knowledge) that supposed to be used to connect given knowledge elements. (Evident information is, for instance, that in Hungary the sun shines from the south, notwithstanding that the order prescribes the lighting of the building. Some lacking meta knowledge could be the way the official searches/applies information in the Code and as he authorizes an object.) Now it is obvious that a Code extended by this knowledge would be much lengthier than the present one. These kinds of facts were used during development as knowledge of the official authorizing civil building activities which completes the knowledge given in text form. These results could not have been achieved without the continuous support of architects working at local authorities and senior experts of the Building Code.

Some decisions can successively be made based on the answers to the questions which are to put in the nodes of the question net (directed graph) that standardizes the working method of officials using the system. As soon as the last question is answered the plan of a building can be authorized in a relatively objective way. The program follows a certain route on the directed graph in a given case the technical content of which is very simple: it always starts from the same node, and input data determine exactly the possible moves. Activities which are to be performed at a node can be given by the same algorithm at any stage. *Fig. 9* shows the record structure of this algorithm. The system was tested by the city councils of Bácsalmás, Budapest XX., Esztergom, Pécs Szeged, Szentes and Veszprém.

5.3. The Rule Based System

If expert systems are to be produced in a large scale, then it would be favourable if the information recorded on different data storing media (like books, standards, human brains, programs, etc.) could be processed without any need of comprehension and adequate rearrangements. In other words, automatic generation of 'rules' from textual knowledge would be desirable. This is, however, a rather difficult question that is still unsolved, even if it is referred so as if any resolution had been reached, SIEGEL (1988). Some real achievements would be obtained if certain restrictive sentence standardizations were to be introduced at the drafting stage of the edition. In our experimental systems sentences were rewritten in a formalized way



a)

Fig. 9a. The algorithm of the activities performed at any node

to get statements that could be transformed to rules. This method is not likely to be useful in large scale expert system production, but it is still suitable in scientific research.

We also have to mention the fact that the method previously discussed was applicable to formulate rules within narrow limits only in the case of the Building Code. The consistent and complete knowledge base of

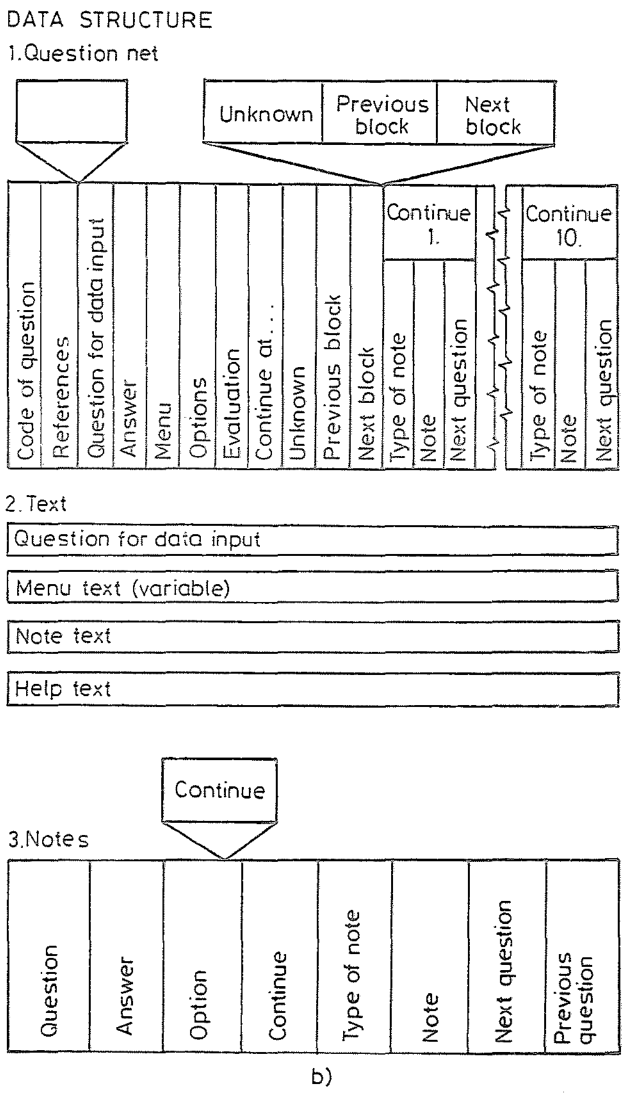


Fig. 9b. The record structure of the activities performed at any node

the intelligent system was extracted not directly from the Building Code itself, but from the decision graph set up on the same basis. We also have to point out that there is no need for dealing with uncertain knowledge in this system. Although, there are some references to uncertain knowledge handling in legal systems as well, DESCHAMPES (1987), the effective prob-

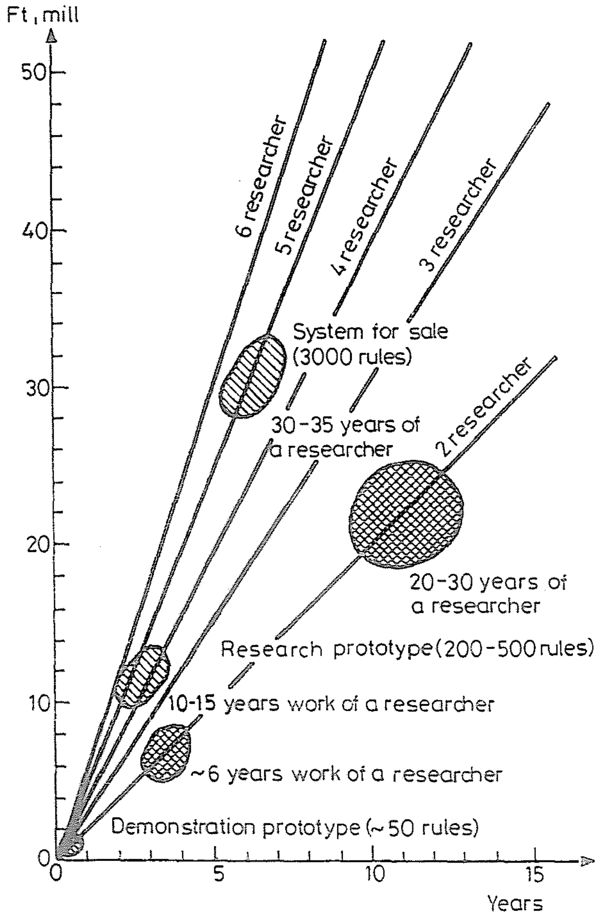


Fig. 10. ES development expenses (HUF) versus time (year)

lem solving of this kind does not allow any uncertainty, i.e. statements of a law are always considered to be firm as in our case.

6. Analyzing Time Horizon and Financial Parameters

Time horizon and financial parameter estimation of expert system production is reliable after developing a great number of systems. We discuss these problems compared to data of references, WATERMAN (1986), and other partial information about systems realized fully or partially (*Fig. 10*).

Demonstration prototypes containing approximately 50 rules need, according to our observations, 2 months work of a researcher (1 person, 2 months) and 200 thousand HUF income (without any investment, expenses of computer use involved).

Research prototypes containing 250–500 rules need 10–15 years work of a researcher (5 persons, 2 years) with 10 million HUF income.

Systems for sale (about 3000 rules, WATERMAN (1986)) need approximately 30–35 years work of a researcher (5 persons, 6 year) and 30 million HUF income.

Our system that also involves an intelligent demonstration prototype version, needed 6 years work of a researcher (2 persons. 3 years) with 6 million HUF income.

Our calculations are performed taking 1 million HUF annual income for every researcher into account that also involves computer use expenses without any hardware investment.

Acknowledgement

The results presented here took a long time work of a team. Our colleagues in the team were Dr. Georgieva Petranka-Zsidó as researcher of intelligent systems, Ernő Nagy who set up the decision graph and Ferenc Csizmás who made all programming work. Our special thanks are due to János Aszalós for his invaluable advices.

Among our other colleagues we shall specially acknowledge the generous help of Edit Sánta-Tóth, Zsuzsa Farkas, Dr. Péter Koch and Dr. András Domán.

Our work was financed by the Department of Development/Ministry of Building and Urban Development and the G1 Program Department/State Office for Technical Development.

References

ASZALÓS, J. (1986): Alkalmasságvizsgálati tanulmány az ÉTI 'OÉSZ szakértői rendszer'-hez, 'Feasibility Study on the Expert System Based on the Text of the Hungarian Building Regulations', (in Hungarian) Építéstudományi Intézet, Budapest, 1986. December.

BONNET, A. (1981): Applications de l'intelligence artificielle: les systémes experts. *RAIRO Informatique-Computers Science*, Vol. 15 (1981). pp. 325–341.

BONNET, A. – HARRAY, J. – GANASCIA, J. G. (1982): LITHO, un systéme expert inferant la geologie du sons-sol. *Technique et Science Informatique*, Vol. 1 (1982), pp. 393–402.

DESCHAMPES, J. – QUENILLET, M. (1987): Systémes experts juridiques: une realité. *7th Int. Workshop on ES and their Appl.* Avignon 1987. pp. 1401–1425.

ES projects in Hungary in MPROLOG, Computer Research and Innovation Center, Budapest, Oct. 1986.

Genesys, an ES Shell, Reference Manual and Tutorial, SzÁMALK, Budapest, 1987.

KOVÁCS, A. (1985): A JIR programrendszerek megépítése, 'Creation of JIR (a Legal Information System)', (in Hungarian) *Információ Elektronika*, Vol. 20 (1985), pp. 188–197.

MPROLOG Documentation, Release 2.1, Logicware Inc.–SZKI, Sept. 1985.

Országos Építésügyi Szabályzat, 'Hungarian Building Regulations', (in Hungarian) Építésügyi Tájékoztatási Központ, Budapest 1986.

RÉDEY, G. – PAÁL, T. (1989): Expert System for Geotechnical Testing of Damaged Buildings, *Acta Technica Acad. Sci. Hung.* Vol. 102 (1–2)(1989), pp. 87–102.

SEPRŐDI, L. – KOVÁCS A. – VÁRKONYI, ZS. (1985a): A Jogi Információs Rendszer szoftverszervezésének folyamata, 'The Software Management Process of JIR' (in Hungarian) *Információ Elektronika*, Vol. 20 (1985), pp. 145–164.

SEPRŐDI, L. (1985b): A JIR koncepciója, 'The JIR Idea', (in Hungarian) *Információ Elektronika*, Vol. 20 (1985), pp. 87–93.

SEPRŐDI, L. (1985c): A JIR továbbfejlesztésének koncepciója, 'Plans of the JIR Improvement', (in Hungarian) *Információ Elektronika*, Vol. 20 (1985), pp. 316–319.

SHIRAI Y. – TSUJII, J. (1987): Mesterséges intelligencia. Alapelvek, alkalmazások, 'Artificial Intelligence. Basic Concepts, Applications' (in Hungarian), Novotrade RT, Budapest 1987.

SIEGEL, P. (1988): VP Expert. Szakértelem szinte ingyen, 'VP Expert. Expertise almost Gratis', (in Hungarian), *Számítástechnika Computerworld,* Vol. 3/1 (1988), pp. 28–29.

VÁRKONYI, ZS. (1985): A JIR programrendszerének felhasználói szintű tesztelése, 'Application Level Testing of JIR', (in Hungarian), *Információ Elektronika*, Vol. 20 (1985), pp. 251–263.

WATERMANN, D. A. (1986): A Guide to Expert Systems, Addison-Wesley Publishing Company, Reading Massachusetts..., 1986.