INFORMATICA, 1995, Vol. 6, No. 2, 181–192

# EXTENSION OF A RULE-BASED SYSTEM APPLICABILITY

Saulius MASKELIŪNAS

Institute of Mathematics and Informatics
2600 Vilnius, Akademijos St. 4, Lithuania

**Abstract.** Rule-based systems are usually interpreted as a shallow expert systems realization tool. The paper analyses how the applicability of production rules can be extended using the proposed rule base structuring discipline. Its main constructions are rule grouping according to elementary aspects of investigation, and decomposition of actions. In addition, the rule cycle construction is used for discrete time simulation tasks. The proposed method is illustrated by 2 applications: the expert subsystem for a database, and the simulator of a water heater.

**Key words:** knowledge representation, rule base structuring, forward and backward chaining, rule-based simulation.

**1. Introduction.** Most frequently rule-based systems (RBS) are used for the realization of expert systems so far, because production rules suit well for the expression of heuristic knowledge. Currently the investigation of possibilities of combining rules with other knowledge representation formalisms (i.e., with frames (Barber *et al.*, 1988), objects (Berndsson, 1994), neural nets (Sun, 1994), etc.) is getting broader. The use of rules in database systems became separate self-dependant research trend, too (Furtado, 1994, Ceri, 1994).

This paper analyses how the usability of rule-based systems in various application areas could be extended by means of the proposed rule base structuring discipline.

In the 2nd section of this paper different views of production rules are analysed. In Section 3 the method of rule grouping at the elementary level of decisions is introduced. The 4th section presents the proposed rule base structuring discipline and outlines its peculiarities. In the following two sections a similarity between rule-based diagnosis and rule-based simulation is shown, and the examples of the concrete rule-based systems realized using the structuring discipline are presented. The last section outlines some conclusions.

**2. Different views of production rules.** The approaches to the production rule interpretation can be classified into the following sets:

(1a) rules as constrained, conditional actions;
(1b) rules as contextual, situational behaviour actions;
(2a) rules as a 'black boxes', shallow reasoning steps, transformational, pragmatic units;
(2b) rules as an expression of causality, influence;
(2c) rules as a form of traductive inference, as transitions in the space of objects, states, decision stages, domains, etc.);
(3a) rules as a means of expressing the classification;
(3b) rules as indicators of qualitative borders, as recognitional units.

In real application areas the problem is how to coordinate qualitative and quantitative data. With the help of production rules the following transitions can be expressed (for their subsequent usage in the reasoning process):

– $Qn \rightarrow Ql$ (i.e., transition from quantitative to qualitative information. This is the most frequent form of production rules. It can be used for expressing abstraction relations between different levels of the represented information);

– $Qn \rightarrow Qn$ (a form of heuristic, shallow, compiled reasoning. Here quality is postulated intentionally, not declaratively);

– $Ql \rightarrow Ql$ (this is the case of pure symbolic reasoning);

– $Ql \rightarrow Qn$ (a form for returning to usual quantitative parameters of the problem area).

Every production rule postulates (in extensional or intentional form) some quality of the possible parameter value area: particular actions in the 'THEN' part of a rule are specified only for the bounded sub-area which is indicated by the 'IF' part of the rule.

The qualitativeness of production rules allows to use them as a control framework for qualitative simulators. I.e., for each qualitatively different interval of parameter values, different equations of parameter calculation are assigned (expressing a different behaviour of the simulated system). The passage beyond the boundaries of the current qualitative interval (i.e., a transition into another qualitative interval of parameter values) means that the behaviour of the simulated system has changed qualitatively. Therefore, from this time moment another set of equations is selected for the calculation of simulation parameters.

Talking about the possibilities of RBS application, the use of forward reasoning rules 'IF $\{conditions\}$, THEN $\{actions\}$' is mentioned most often. However, treating RBS only as a tool of shallow, heuristic reasoning, their potential application scope is narrowed a little. As shown in Southwick (1990), the backward reasoning RBS differ essentially from the forward reasoning ones. Knowledge bases organized in such a manner express a gradually detailed structure of the decision process. Therefore, they are more declarative than forward chaining RBS.

The form of a backward chaining RBS production rule is very similar to that of the Horn clause (used in the logic programming language PROLOG). A declarative and procedural interpretation of Horn clauses can be used to backward chaining production rules. In the case of simulation tasks a third interpretation mode – the process interpretation – is indicated in Futo (1988). It shows that a left-side process of the Horn clause consists of its right-side processes (actions, operations, etc.), which, in their turn, are specified in the following clauses. This interpretation can be used for backward chaining production rules as well.

The comparison of production rules with other knowledge representation formalisms, i.e., first-order predicate logics, semantic nets, and frames (according to the axes of inference possibilities, formality, possibility of additions, notational adequacy, and semantic declarativity) is given in Butleris (1987).

**3. Proposal of rule grouping.** General ways of coping with the complexity in computer software organization are the following: encapsulation (introduction of abstract data types), specialization (or module separation), introduction of hierarchy (or systematization), implanting of discipline (or limitation, simplification of possibilities).

For the ordering of rule bases the introduction of hierarchy (Bharadwaj and Jain, 1992) and module separation techniques are used most often. The latter allows to split the problem space into more understandable pieces; it tends to simplify the rule construction, debugging and the maintenance by alleviating context issues, etc. (Fickas, 1989).

The moduling of production rule bases can be organized at different structural levels: separating rule packages as programs (Fickas, 1989), dividing the rule base into decision contexts (knowledge islands) (Rappaport and Gaines, 1990), object-oriented modules (Levine *et al.*, 1988), or conceptual clusters

(Cheng and Fu, 1995).

In our work production rule grouping at the lowermost level (i.e., at the level of elementary decisions) is used. One group of production rules expresses one semantically elementary aspect of problem area investigation. The values of a parameter or of a combination of several parameters are verified in the 'IF' part of rules of a separate rule group. (E.g., to evaluate possible delays in the execution process of plan, production rules are prepared for a normal case, for small, average and great delays). Thus, the 'IF' part of every rule of a rule group defines a qualitatively different case, and the corresponding actions to it (and/or the direction of subsequent needed actions) are specified in the 'THEN' part of that rule.

The net of rule groups is a declaratively expressed context for actions of every rule. In addition, such rule grouping is convenient for Top-Down design of rule bases, and fits well for more exhaustive explanation purposes.

The elementary rule grouping can be used both in forward chaining rule-based systems and in backward chaining ones. The rule grouping in a forward chaining RBS can be organized, e.g., introducing an additional variable $R\_G$ (which is used as a rule group identifier) :

$$\text{IF } (R\_G =< \text{rule group} >, \quad \{\textit{rule conditions}\}),$$
$$\text{THEN } (\{\textit{rule actions}\}, \quad R\_G :=< \text{next rule group} >).$$

In this case, an ordered set of rule groups with one rule in each of them behaves as a set of backward chaining rules; and both forward and backward chaining can be used side-by-side.

On the other hand, in a backward chaining RBS a rule group can be expressed as the mega-rule:

$$A \text{ IF } ((a_1? \ \& \ A_1) \vee (a_2? \ \& \ A_2) \vee \ldots \vee (a_n? \ \& \ A_n)). \qquad (1)$$

Here: when $a_i$ = 'true' $(i = 1, 2, \ldots, n)$, then $A_i$ is executed only in the mega-rule (1); and if $a_i$ = 'false', then the $a_{i+1}$ is tested next. So, the rule group $A$ consists of a set of conditional actions which are the forward chaining production rules: 'IF $\{a_i\}$, THEN $\{A_i\}$'.

The introduction of such kind of rule grouping gives a possibility of eliminating drawbacks of used backward chaining vs. forward chaining rule-based system, appropriately combining the qualities of both of them.

**4. Common structuring method for rule bases.** The use of rule grouping constructions (1) in a backward chaining RBS expresses the execution of pure classification (diagnosis). In order to build rule-based systems for other tasks, too, one more construction is needed for decomposition purposes:

$$B \text{ IF } B_1 \ \& \ B_2 \ \& \ \ldots \ \& \ B_m. \tag{2}$$

Using it all the parts of the rule base can be expanded: 'IF' part of rules (introducing preliminary calculations, information gathering, etc.), 'THEN' part of rules (recursively, as well), rule base control constructions.

The subcase of (2):

$$B_m = B \tag{3}$$

is also allowed here: it means restarting (repeated execution) of rule (2). In addition, each $A_i$ and $B_j$ of (1) and (2) in its turn can be $A$, or $B$, or an elementary action, or logical constant 'true'(i.e., '0' action).

Consequently, construction (1) expresses case-separating evaluations, construction (2) denotes the hierarchical decomposition of actions, and construction (3) allows to organize iterative processes. Their combination can be used as a building discipline for RBS knowledge bases: all its components can be organized using only these blocks.

A formal expression of the rule base structuring discipline is as follows:
Rule Block (RB) ::= {Rule Group (RG), Sequence of Actions (SoA), Rule Cycle (RC), Elementary Action (EA)}
Rule Group ::= <RG name> IF $(a_1? \ \& \ A_1) \vee (a_2? \ \& \ A_2) \vee \ldots \vee (a_n? \ \& \ A_n)$,
here: $a_j?$ – 'IF' part of forward chaining rule $j$,
$A_j$ – 'THEN' part of forward chaining rule $j$, which is RB by itsel
Sequence of Actions ::= <SoA block name> IF $B_1 \ \& \ B_2 \ \& \ \ldots \ \& \ B_m$;
here: $B_i$ is Rule Block.
Rule Cycle ::= <RC name> IF $C_1 \ \& \ \ldots \ \& \ C_m \ \& \ c_y? \ \& \ \mathit{Repeat} \vee$ 'T',
here: $C_i$ is Rule Block,
$c_y?$ is a tested condition of cycle repetition,
$\mathit{Repeat}$ – control construction for recurrent handling of $C_1, \ldots,$
'T' is the logical constant 'true' (for normal exit from the cycle).
Elementary action ::= {Assigning of parameter value; Preparing of control/ display videogramme; Condition testing; Execution of arithmetical calculations or a special function; etc.}

Depending on what blocks are used at the top levels of decision structure (i.e., RG or SoA), a rule base can be organized in 2 different manners: as a gradual descending or as a sequence of stages.

In the case of gradual descending, the constructions "Sequence of Actions" are avoided to use in structuring a rule base; the decision process has a gradual classification form; every next rule group belongs to a lower level with respect to the previous one. The amount of needed variables and rule condition tests is minimal in this case, the decision process is compactly expressed.

In the case of a sequence of stages, the SoA constructions are intensively used. That allows to decrease the complexity of a rule base (decreasing the total depth of decomposition), to make the rule base debugging, modifications and maintenance easier.

**5. Qualitative simulation by RBS.** The use of the proposed rule base structuring method allows to organize a discrete-time simulation which combines strict qualitative evaluations with a parameter value change (in one simulation step) calculations. The latter can be organized using numeric discrete change equations or certain qualitative simulation ones.

All the rules used for simulation system organization can be classified into the following types:

– the arrangement of a general control strategy and the execution of state change calculations;

– the expression of the simulated system and manipulations with it, i.e., estimation of the simulated system state change in one simulation step, evaluation of dependencies, conditional changes and interaction of subsystems;

– the determination of parameter values and dynamic scheme components of the simulated system; the representation of the main parameter values and of the simulated system scheme on the user's control screen;

– the organization of an interactive connection with the user (which controls the simulation process, introduces/removes simulated faults, restarts simulation process, etc.);

– the semantic control and normalization of user's messages.

Using the same formalism, both the behaviour of the simulated problem area object and its control unit can be expressed. A backward chaining RBS suits directly for the simulation task solving. Meanwhile, a forward chaining RBS can be used for this purpose only in combination with frame-based, object-oriented or other module separation technique, without which it remains only a tool of realization of shallow, compiled reasoning systems.

The main parts of the simulation cycle are the following:

– the information is given to the user about the simulated object current state; the user's control commands and indications of occurred/removed simulated faults are received. (That means an interactive participation of the user in the simulation process, allowing a "what-if" analysis during the simulation time);

– qualitative intervals (qualitative values of parameters) are established for quantitative parameter values of the simulated object at the current moment $t_i$, $i = 0, 1, 2, \ldots$ of simulation time;

– the corresponding equations for parameter calculation are selected for the available set of qualitative values;

– using parameter values (of the current time moment $t_i$) and the selected equations, the values of the simulated object parameters are determined for the next moment $t_{i+1}$ of simulation time. Eventually, $t_{i+1}$ becomes the current moment of simulation time, the simulation cycle proceeds from the beginning (for the next simulation interval).

Two main phases of the simulation are: a qualitative evaluation of the simulated system state (with selection of appropriate equations for it), and calculation of the simulated system state change in one next step (with the use of selected quantitative or qualitative equations). So, a simulator can be viewed as a diagnostic, evaluation system with feed-back links and additional state-change calculations of the simulation process. Such an approach is especially useful in problem areas with many different qualitative states and/or with complicated changes. Other qualities of it are:

– convenience for realization of research prototypes;

– convenience for modification of simulation system components, subsystems, parameters;

– faster and simpler debugging of the simulation system because of its hierarchical organization;

– good visualization capabilities, applicability in getting acquainted of novices with the investigated problem area.

## 6. Examples of applications

**6.1. Rule-based subsystem for a large data base.** The first application area of the rule grouping technique was the prototypic production rule-based expert system, tightly coupled with a data base system and with the declarative description of DB search and parameter calculation.

A more distinctive component of this application is the work area (WA). It is organized as a tree structure whose nodes are the objects (or situations) under investigation. The WA objects of the next lower level are generated automatically for the current WA object when it turns out that the characteristics of more than one application area object represented in DB (e.g., some executives, some themes, etc. in R&D) meet the requirements of the current situation investigation stage. Thus, the situation under investigation (i.e., the current WA object) is divided into sub-situations (later on, maybe, into sub-sub-situations and so on); then each of these objects is analysed separately. The tree of WA objects is processed with a "deep-first" strategy, (when the "breadth-first" strategy is used for gathering the requested DB data).

An example of the query to the rule-based subsystem of the database is: "Ascertain the situation of the enterprise and give an enlarged estimation for <dispatcher> for ...themes (with ...ponderability) processed by ...departments for ...jobs. For critical situations: analyse detailed and foregoing works, determine tasks and offer techniques of solution".

Other features of this application system are presented in Maskeliūnas (1989).

**6.2. Water heater rule-based simulator.** An example of a water heater simulator was selected in our case for comparison of the proposed simulation method with the usual qualitative simulation one. (The main qualitative border of the simulated water heater is thermostat temperature: when the outlet water temperature exceeds it, the power supply to braziers is interrupted). The realization of the water heater simulator by means of QSIM (i.e., the well known qualitative simulation tool) is described in Dvorak *et al.* (1990). In our case a backward chaining rule-based system was used for that.

The main peculiarities of the realized water heater simulator are the following:

– the simulation proceeds by steps. In a separate step the state change in each subsystem (during the elementary simulation time interval) is calculated. This change depends on: 1) the previous state of a subsystem, 2) the influence of other subsystems and environment on this subsystem, 3) a transition of parameter values into qualitatively different intervals of values, and 4) user's commands and specified parameter values. The last source of changes allows to investigate the consequences of various combinations of parameter values and accidents in a simulated system;

– quantitative values of parameters are estimated by approximate calculation equations depending on qualitative restrictions. (The coefficients of calculation equations are defined in the tuning phase of simulation system design. When the simulation continues qualitative changes can arise because of process evolution and/or because of user's control commands and introduced faults);

– for the expression of transfer functions the propagation functions (Trave-Massuyes, 1990) are used in our case. (I.e., the influence of every brazier onto water temperature in the heater is calculated by the formulae of such type: $Y[t_i, t_{i+1}] = A \times X(t_i) + B \times X(t_{i-1}) + C \times X(t_{i-2})$, where $A + B + C = 1$ and $X(t_i) = 1$, if the brazier is functioning during time interval $[t_i, {}_{i+1}]$, and $X(t_i) = 0$, otherwise).

The main production rule of the simulator is as follows:

```
Simulation cycle
  IF Parameter values tracing
  & Simulation supervision and control
& ('Continue'?)
& step := step + 1
& Intensity of the current
& 1 brazier effect
& tm_braz1 := .7 × braz1_ + .2 × braz1_p + .1 × braz1_pp
& 2 brazier effect
& tm_braz2:= .4 × braz3_ + .4 × braz2_p + .2 × braz2_pp
& Water getting cooler due to the flow out of water
& Self-cooling of water
& tm:= tm + tm_braz1 + tm_braz2 – tm_fl_out – tm_cooling
& Outlet water tm ⩽ inlet water tm
& Processing of the water boiling case
& Processing of tm – thermost_tm relation change
& Restart
∨ ('Stop'?)
```

Owing to the realization of this experimental simulation system some disadvantages and advantages of the used simulation method were revealed. Its shortages are the following:

– in one simulation step only one next state (corresponding to the current state of the simulated object) is established;

– the new state estimated of the simulated object may be qualitatively the same as the previous one (therefore, qualitative changes in the simulated system are expressed more distinctly in the usual qualitative simulation systems: the result of simulation is a sequence of only qualitative change moments in them);

– a necessity to know in advance all possible directions of the use of problem area equations (describing the simulated object) and to prepare a separate part of calculations for each of them.

The qualities of the realized simulation are as follows:

– estimation of relative durations of processes in the simulated system (this decreases the uncertainty of the simulated process evolution);

– readiness for user's active participation in the simulation process: the possibility of entering control decisions, simulated faults, changes in simulated system parameters (at the beginning of each simulation step) and of establishing their influence on the following simulation process;

– the possibility of restarting the simulation (from the indicated step of already executed simulation) with defined changes, i.e., the possibility to evaluate all different cases of simulated system evolution which are of interest to the user;

– such a simulator is rather declarative, with separated different levels of abstraction. So, it is suitable to prototype development, frequent modifications of the simulated system.

More detailed description of this system is given in Maskeliūnas (1993).

The simulator of the same type was realized to analyse rinsing bath (as a component of plating processes) design alternatives. Its description is presented in Šalkauskas and Maskeliūnas (1994).

The proposed rule-based structuring method was used in the simulation of competition of service enterprises. In this case two additional important features were introduced: 1) variable duration of the simulation step; 2) combination with the global optimization.

1. The simulation process here is a combination of three asynchronous sub-processes (i.e., the income flow of customers, service at the 1st and the 2nd enterprise). The checked time moments are when a qualitative change occurs in any of the simulated subsystems. The duration of a simulation step varies, dependent on the concrete calculated sequence of qualitative changes.

Such an improvement allows to realize any usual qualitative simulation, (Trave-Massuyes, 1990) not only constant step simulations. Thus, qualitative simulations can be realized by the rule-based system (i.e., by the available expert system shell), without applying special qualitative simulation tools.

2. The search for the optimal strategy (cost&price positioning) was simulated. The rule-based simulation method was combined with the Monte Carlo global optimization method, realized using the same rule-based system.

The attained results of the competition simulation are presented in Tiešis *et al.* (1994). Such a simulator is convenient for obtaining principal operations of the simulated system, for a comparative investigation of different alternative cases, educational and training purposes.

**7. Conclusions.** After the period of successful and broadening usage of rule-based systems in various practical applications some shortages of RBS were established. I.e., shallowness of expressed reasoning, and rather a complicated testability, maintainability in the case of larger rule bases. In order to solve those problems the formalism of production rules is combined with the means of deep knowledge representation, various module separation approaches to rule bases are used.

In our case, the mentioned problems are solved by means of the rule base structuring discipline, applying the set of knowledge representation constructions: elementary aspect rule group, decomposition unit, and rule cycle (the last of them can be used for discrete-time qualitative simulation, monitoring etc. tasks). Such a method allows the design of hierarchical well structured rule bases in a Top-Down and prototypic manner; with the help of it both diagnostic and discrete qualitative simulation tasks can be realized using the same backward chaining RBS and the same knowledge representation framework.

Proceeding with this work, the presented method of rule base structuring will be used in ecosystem qualitative analysis, decision support, and other tasks.

## REFERENCES

Barber, T.J., Marshall, G. and Boardman, J.T. (1988). Tutorial – a philosophy and architecture for a rule-based frame system: RBFS. *Eng. Appli. of AI,* **1**, 67–86.

Berndsson, M. (1994). Management of rules in object-oriented databases. In J. Bubenko et al. (Eds) *BalticDB'94 Workshop proceedings,* Vol. 1. Mokslo Aidai, Vilnius. pp. 78–85.

Bharadwaj, K.K., and Jain, N.K. (1992). Hierarchical censored production rules. *Data & Knowledge Engineering,* **8**(1), 19–34.

Butleris, R. (1987). Knowledge representation model for expert system of complex discrete–continuous technological object control. *Cand. of Techn. Sc. degree thesis,* Kaunas Polytechnic Institute, Kaunas. 197 pp. (in Russian).

Ceri, S. (1994). Active database systems. In *PUC-Rio DB Workshop on New Database Research Challenges Proceedings, VLDB Tutorial Programme,* Pontificia Universidade Catolica do Rio de Janeiro. pp. 35–52.

Cheng, Y., and Fu, K.-S. (1985). Conceptual clustering in knowledge organization. *IEEE transactions on Pattern Analysis and Machine Intelligence,* **PAMI–7**(5), pp. 592–598.

Dvorak, D.L., Dalle Molle, D.T., Kuipers, B.J. and Edgar, T.F. (1990). Qualitative simulation for expert systems. In *11th IFAC World Congress "Automatic Control in the Service of Mankind",* Vol. 7. Preprints, Tallinn, Estonia. pp. 204–209.

Fickas, S. (1989). Design issues in a rule-based system. *The Journal of Systems and Software,* **10**(2), 113–123.

Furtado, A.L. (1994). Rule-based databases. In *PUC-Rio DB Workshop on New Database Research Challenges Proceedings, VLDB Tutorial Programme,* Pontificia Universidade Catolica do Rio de Janeiro. pp. 19–34.

Futo, I. (1988). AI and simulation on PROLOG basis. In C.A. Kulikowski, M.R. Huber and G.A. Ferrate (Eds.), *Artificial Intelligence, Expert Systems, and Languages in Modelling and Simulation, Proceedings of the 1st IMACS Symposium,* Barcelona, Spain, North-Holland, Amsterdam. pp. 15–20.

Levine, R.I., Drag, D.E. and Edelson, B. (1988). Object-oriented expert systems. In *A Comprehensive Guide to AI and EXPERT SYSTEMS: Turbo Pascal Edition,* McGRAW–HILL, New York. pp. 199–204.

Maskeliūnas, S. (1989). Supplement of database system with knowledge components. In P. Beregy et al. (Eds.), *Proc. of the 12th Int. Seminar on DBMS,* Book 1, Sc.-Prod. Assoc. Tsentrprogrammsystem, Kalinin. pp. 94–105.

Maskeliūnas, S. (1993). The use of a rule-based system for qualitative reasoning. In M. Singh and N. Piera (Eds.), *Qualitative Reasoning and Decision Technologies,* CIMNE, Barcelona. pp. 701–710. (or In *SCAI–93 proceedings,* IOS Press, Amsterdam. pp. 50–59).

Rappaport, A.T., and Gaines, B.R. (1990). Integrating knowledge acquisition and performance systems. In S.J. Barter and M.J. Brooks (Eds.), *AI'88, 2nd Australian Joint Artificial Intelligence Conference,* Adelaide, Australia. *Proceedings, Lecture Notes in Artificial Intelligence,* subseries of L. n. in Computer Science, Vol. 406, Springer–Verlag, Berlin. pp. 307–326.

Southwick, R.W. (1990). A reason maintenance system for backward reasoning system. In Z.W. Ras, M. Zemankova and M.L. Emrict (Eds.), *Methodologies for Intelligent Systems,* Vol. 5, Elsevier Science Publishing Co. pp. 102–109.

Sun, R. (1994). Robust reasoning: integrating rule-based and similarity-based reasoning. *The Computists' Communique,* 4(25),: sun.aij.ps.Z in pub/tech-reports on aramis.cs.ua.edu. 1–52.

Šalkauskas, M. and Maskeliūnas, S. (1994). Rinsing bath model for plating. In *Multiple Paradigms for Artificial Intelligence, STeP–94 Proceedings,* Turku, Finland. pp. 264–266.

Tiešis, V., Maskeliūnas, S. and Skurikhina, M. (1994). The investigation of two competitive systems. *Informatica,* 5(1–2), 211–230.

Trave-Massuyes, L. (coordinator) (1990). Qualitative reasoning: methods, tools and applications, *MQ&D Project, G.R. 'Automatique/AI',* L.A.A.S.-C.N.R.S., Toulouse. 85 pp.

Received March 1995

**S. Maskeliūnas** is a researcher at the Knowledge Based Systems Department of the Institute of Mathematics and Informatics, Vilnius, Lithuania. His research interests include knowledge representation, qualitative reasoning, discrete time simulation, expert systems.

## PRODUKCINIŲ TAISYKLIŲ SISTEMOS PANAUDOJIMO GALIMYBIŲ IŠPLĖTIMAS

Saulius MASKELIŪNAS

Paprastai produkcinių taisyklių sistemos yra laikomos seklių ekspertinių sistemų realizavimo priemone. Šiame straipsnyje nagrinėjamas produkcinių taisyklių panaudojimo išplėtimas siūlomos produkcinių taisyklių bazės struktūrizavimo disciplinos pagalba. Šios disciplinos pagrindinės dedamosios yra taisyklių grupavimas pagal elementarius tyrimo aspektus ir veiksmų dekomponavimas. Be to, diskretinio laiko modeliavimui, monitoringo ir kt. užduotims dar naudojama taisyklių ciklo konstrukcija. Siūlomo metodo iliustravimui pateikiama duomenų bazės ekspertinės posistemės ir vandens šildytuvo modeliavimo sistemos pavyzdžiai.