БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ . BULGARIAN ACADEMY OF SCIENCES

ПРОБЛЕМИ НА ТЕХНИЧЕСКАТА КИБЕРНЕТИКА И РОБОТИКАТА, 55
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS, 55

София . 2005 . Sofia

Defeasible Inference in Intelligent Agents of BDI Type*

Stefan Kojnov, Vladimir Jotsov, Vassil Sgurev

Institute of Information Technologies, 1113 Sofia

1. Introduction

The method with intelligent agents in multi-agent systems (MAS) in comparison with the traditional approaches enables the increase of efficiency in information protection, including their adequacy, failure resistance, destruction resistance, universality, flexibility, etc. [1] The success of such systems is due to a great extent to the distributed way of solving the problems and the search for an approximate solution instead of an optimal one [2].

A specific feature of the distributed decision making in MAS is the necessity every agent to make decisions in lack of complete information concerning the environment and the other agents [2]. The lack of information is overcome using techniques like messages exchange and the use of protocols. The co-operation of the agents has its specific problems connected with the solution of the conflicts occurring among them [3, 4].

The present paper discusses the problems of solutions annulment in multi-agent systems – one of the leading directions for solving conflicts in these systems. At first a short review is made of previous works in the area of MAS. After that a system of notions is introduced facilitating the solutions annulment in the area together with the corresponding logical formalism. Some examples are given of agents. In which the approach considered cannot be applied. The paper ends with some inferences.

* This paper is supported by Grant No TH1408/04 of National Fund for Scientific Research of the Bulgarian Ministry of Education and Science.

2. Apriori research. Basic notions in BDI architectures

A scheme is presented in [5] and [6] of an intelligent agent with features of BDI architecture, shown in Fig. 1. The principle of the functional units is applied to it or FUNs (acronym of Functional UnitS) [7, 8, 9, 10], which on its turn consist of functional units at a lower level, etc.

This scheme is in fact of a layered type (each layer is processed and it transmits the information available in it *in parallel* and *independently* on the other layers). The information in it is in three parallel flows of the three possible models (plus the ensemble model for ensemble learning) and the three (with the block for voting or ensemble learning – four totally) parallel information "tracks" – coordinatewise / structural, probabilistic and regressional – practically end in the voting block with ensemble training.

The characteristics, which relate the model with BDI (the abbreviation is an acronym of Belief-Desire-Intention) architectures is the structuring in four groups of highest level: a block for environmental description, a block of strategies, a block of tactics and a block for the undertaken actions.

•*Block for environmental description* (*beliefs*). It consists of a subgroup for analysis of the environment followed by a subgroup of the possible models based on the analysis done.

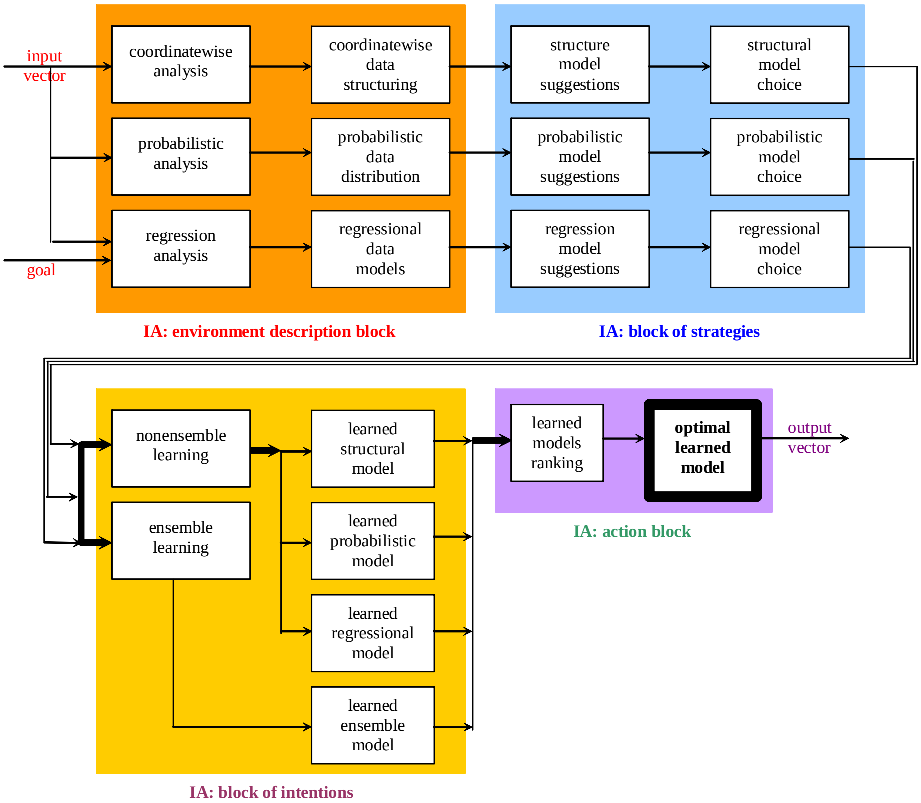

Fig. 1. Intelligent agent of a layered type and BDI architecture

•*Block of strategies* (*desires*). It is built by the subgroups of the possible suggestions for action followed by the selection of a definite suggestion.

•*Block of tactics* (*intentions*). It also consists of two subgroups. The first subgroup consists of the sub-blocks for ensemble and non-ensemble learning. The second subgroup is built by trained for the current suggestion models, selected in the previous functional units.

•*Block of the undertaken actions* (*actions*). It includes two sub-blocks – a block for learned models ranking and a block for the selection of the optimal in the sense of the application given model.

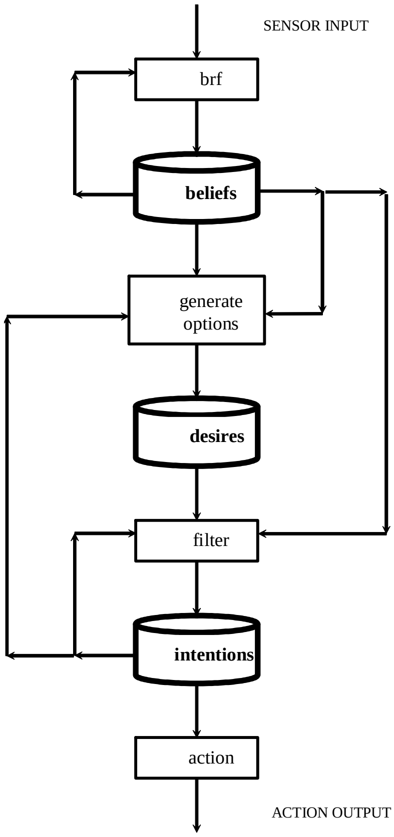

Fig. 2. BDI architecture

The functional units thus presented of the highest level correspond to the main parts of a typical BDI architecture given in Fig. 2. The diagram shows with bold lines the corresponding data bases, and with thinner – the functions of revising the plans (brf), for options generation (options generator), for filtering the intentions (desires filter) and action selection (action selection), as well as the sensor input (sensor input), the action output (action output) and the transition edges. On its hand the function for action selection has the algorithm shown in Fig. 3. In it γ denotes the speed of environmental change. At small γ the environment does not change quickly and the brave agents are typical for such environments. Since the role for decision making grows with the increase of γ, at large γ the environment is often altered and precautious agents are typical for such environment [11].

Function Action(p:P):A	{ BDI architecture action choice function }
Begin {*Action*}	
B : = br f(B,p)	{ $\forall$B $\in$ γ(Bel), B $\subseteq$ Bel; Bel – set of all possible plans }
D : = options(D,I)	{ $\forall$D $\in$ γ(Des), D $\subseteq$ Des; Des – set of all possible strategies }
I : = filter(B,D,I)	{ $\forall$I $\in$ γ(Int), I $\subseteq$ Int, filter(B,D,I) $\subseteq$ I $\cup$ D; Int – set of all possible tactics }
Return execute(I)	
End {*Action*}	

Fig. 3. BDI architecture action choice function algorithm

It can be seen that the presented diagram belongs to BDI architecture only in its main features maybe of an abstract agent or more likely of a *specifically-autonomous agent of reactive type*. It is reactive because it is influenced in its work only by its own input information (perceptions) [11]. The agent is specifically autonomous because it does cooperate with the other agents in order to achieve its goals and because it interacts completely independently with the environment – it makes decisions on its own, based on models which it develops and checks (in [11] from the definition of the notion autonomy on page 584 and from the shorter explanation on page 387 it is understood that autonomy does not exclude communication with other agents), instead it develops and verifies different data models that are later subjected to ensemble voting.

The inconvenience in operation with such type of agents is a sequence of their non-sociality, as far as they do not account the action of the remaining agents. That is why their application at the lower levels is recommended (in the more complicated agents) – probably as *intelligent actuating mechanisms*.

A main shortcoming of this expose is the absence of interaction and solution of the conflicts among the agents. The following discussion concerns the construction of a system of notions and methodology aimed at the solution of conflicts among the agents by annulment of the inappropriate solutions.

3. Defeasible inference of conflicts among agents

Let us imagine that certain agent acts together with other agents in some form of joint work. In this case there exists some probability for conflicts among the agents based on their current possibilities and intentions to reach their aims.

The occurrence of conflicts is directly connected with the defeasible inference of the actions of some agents which up to the moment due to different reasons cannot realize their goals. The defeasible inference of the agent's actions may be regarded at several levels: a level of the agent as a single unit, the level of *actions*, the level of tactics (*intentions*), the level of strategies (*desires*) and the level of plans (*beliefs*). At that we see the following mutual dependence of these defeasible inference levels (Fig. 4) where with DI we denote the defeasible inference itself (DI is an acronym of Defeasible Inference); the scheme does not show the source of the DI information inspired by the agent's action defeasible inference. The pattern for this dependence is the immediate cause for the defeated action (the information stream starts from the beliefs, it goes through the desires and the intentions and it ends with the initiated action of the agent (except for the feedback from the intentions to the desires) – see Fig. 2 but defeating an action spreads in the inverse direction because it acts like a backtracking).

The following below text is dedicated consecutively to defeating agent's action followed by defeating tactics, strategies and plans of an arbitrary agent with a BDI architecture.

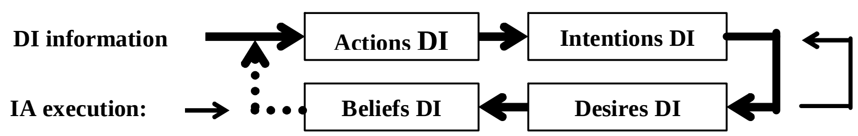

Fig. 4. Information streams for execution and defeasible inference

3.1. Defeating agent's action

3.1.1. Criteria and conditions for conflict solution

Criteria (factors) for conflict solution

They are included in the set {R, I, P, Π, T }, where
R – resource(s),
I – indicators(s),
P – priority/ies,
Π – probability/ies to satisfy the requirements,
T – technological necessity;
The elements of this superset (multiset) themselves are subsets.

Factors I, P and Π are related to the so called *condition for active expectance* (see below).

Besides they may be feasibility factor(s) and defeasibility factor(s). Whether some factor for some agent belongs to one of these two groups depends on whether this factor is respectively better or worse than the factor with the closest meaning of any of the other agents in the set of agents. Let us have two agents A_a and A_b with their corresponding factors for every one of the two agents $A_a = \{R_a, I_a, P_a, \Pi_a, T_a\}$ and $A_b = \{R_b, I_b, P_b, \Pi_b, T_b\}$. Besides let us have the following relations: $R_a>R_b$, $I_a>I_b$, $P_a>P_b$, $\Pi_a>\Pi_b$, $T_a>T_b$.

Then we may say that each element in the set for A_a satisfies the requirement for feasibility and v.v.: every element in the set for A_b satisfies the requirement for defeasibility; as a whole agent A_a satisfies the requirements (factors) for feasibility and agent A_b satisfies the requirements (factors) for defeasibility.

Table 1 presents the classification of the factors according to the agent's (de)feasibility and their significance ranking according to the plans' (de)feasibility denoted by ***P***.

Table 1. Classification and ranking of factors for conflict solution

Feasibility factor(s)		Defeasibility factor(s)	
P	Feature	***P***	Feature
1	Technological necessity	10	Technological indifference
2	Bigger probability / ies	9	Smaller probability / ies
3	Higher priority / ies	8	Lower priority / ies
4	Better indicator(s)	7	Worse indicator(s)
5	Enabled resource(s)	6	Disabled resource(s)

The authors relate the feasibility factor to the case when the agent's characteristic *enables* the goal to be achieved in comparison with the respective characteristic of other agents. On the contrary, the defeasibility factor is related to the case when the agent's characteristic *disables* the goal to be achieved in comparison with the respective characteristic of other agents. Analogously the technical necessity here is treated as a requirement for agent's *inclusion* in the information stream; the technical indifference is treated as a requirement for agent's exclusion from the information stream. The ranking from Table 1 shows that:

• feasibility factors suppress defeasibility factors in cases when the agent has to perform some action to achieve its goal;

• defeasibility factors suppress feasibility factors when the agent must defeat its action.

Besides table 1 shows that defeasibility factors are ranked in the inverse sequence.

In cases when one factor is presented by more than a single value (i.e. by a set or vector of values) then some policy must be applied for the fully evaluation of the multiple factor (some generalized measure or the prevailing type of the homogeneous values). For example let the factor be a vector or a matrix of some dimensionality. Now the policy for fully estimation may be the usage of a suitable scalar meaning that describes definite characteristics of the multidimensional matrix. If the factor is a set of vectors or matrices then a set of estimates may be used that in turn may be reduced to some single measure (e.g. the set of estimates may form some estimation vector and the single measure may be its absolute value, polar angle, etc.).

The conditions for conflict solution reflect the possible states of some agent. They can be grouped in three basic groups and each group depends directly on its predecessor: condition for conflict, condition for (de)feasibility and wait condition (active or passive).

Condition for conflict. The goal is achieved or caught by another agent. This condition always precedes all other conditions (the logical formulation is introduced in the next division).

Conditions for (de)feasibility are:

- **Condition for feasibility**. It is defined by the active for the performance (and enabling the execution) feasibility factor.
- **Condition for defeasibility**. It is defined by the active for the performance (and disabling the execution) defeasibility factor.

For example let two factors Φ_1 and Φ_2 possess a mutual relation $\Phi_1 > \Phi_2$. Then factor Φ_1 is feasible and factor Φ_2 is defeasible (also see item 'Criteria (factors) for conflict solution').

Wait conditions (active or passive) are:

- **Active-wait condition**. The objective feasibility factor is inside the range between the upper and the lower bounds of the value. They are set by the most appropriate agent and the most unsuitable agent. Logically this means that the agent is able to 'push out' the agents with worse indicators.
- **Passive-wait condition**. This condition always holds when the agent is in a want of resources and if there is a technological necessity of the agent's performance.

The already postulated conditions that form the state space of the agent form the basis of solving conflicts between the agents.

3.1.2. Logical formulation of conflicts’ defeating

Classical binary logic is related to formalizations of strictly correct (formal) arguments. Still the object field that is the background for the basic concepts and conclusions possesses an incomplete, inaccurate, contradictory, and frequently variable information [12-18]. So there is a necessity to use and develop new non-classical methods for formalizing intelligent processes and information technologies.

At present we have a mighty big variety of different non-classical logics [13, 14, 18]. Yet the methods for application of these logics in tangible problems are poorly developed. Besides the potential of these logics (e.g. the *K*-valued logics) does not perfectly satisfy the necessities that originate during the elaboration of intelligent systems and technologies.

The bibliography defines the notion of a goal with plans and their clauses [19]. Every clause of plan is of the form

$$\text{goal} : B_1\ .\ .\ .\ B_n \leftarrow S_1;\ .\ .\ .\ ;\ S_m,$$

where every B_i is a belief and every S_i is an action or a subgoal.

Multiple-valued and probabilistic logics are introduced in MAS based on the concept of classes forming the model – S_i. The classes are related to one another with a dependency of the type ‘ancestor-successor’. The differences between class S_{i1} and the other successors of the common ancestor S_i are created by elements $x_{s_i;1}$, $x_{s_i;2}$, ... The result of applying a three-digit logic leads to values “true”, “false” and “uncertainty”. According to [20] defeasible inference is based on the exclusion $E(C, A_k)$ where A_k is x_k or $\neg x_k$, and the conclusion is formed when all successors are of the form $A'_1 \wedge A'_2 \wedge ... \wedge A'_n$; A'_k may coincide with A_k or it may include (using a disjunction) A_k and analogical terms for other variables.

Let rules of a Horn type describe some domain $B \leftarrow \underset{i \in I}{\wedge} A_i$. The extended inference models with exclusions were introduced and generalized in formalized ones in [21, 22] in the following form:

$$\frac{B \leftarrow \overset{z}{\underset{i=1}{\Lambda}} A_i, C, E(C, A_k), \neg A_k \leftarrow C}{B \leftarrow A_1 \wedge A_2 \wedge ... A_{k-1} \wedge \neg A_k \wedge ... A_z},$$

$$\frac{C,\ B \leftarrow \overset{z}{\underset{i=1}{\Lambda}} \mathrm{A}_i, E(C, A_k)}{B \leftarrow A_1 \wedge ... A_{k-1} \Lambda A_{k+1} \Lambda ... A_z},$$

$$\frac{C,\ B \leftarrow \overset{z}{\underset{i=1}{\Lambda}} \mathrm{A}_i,\ \ E(C, A_k)}{B \leftarrow A_1 \wedge ... A_{k-1} \Lambda (A_k \mathrm{V} C) \Lambda A_{k+1} ... A_z}.$$

It is clear that the exclusions are a kind of special-rules inclusions with their effective fields. For example the first formula means that if there exists an exclusion $E(C, A_k)$ that is related to one of the rules with a conclusion B and A_k is its effect then the conjunct A_k must be replaced by $\neg A_k$; when C is not 'true' then the corresponding replacement is impossible (application of the Modus Ponens rule means that the relation between B and $\neg A_k$ leads to a formal logical contradiction).

Therefore the formation of exclusions of the type $E(C, A_k)$ may lead to a contradictory result that is provoked by an incompleteness in the description of the object field. In the case when C is true then the exclusion $E(C, A_k)$ includes this meaning in the conjunct A_k to defeat the meaning of the last conclusions. The result is that A_k is replaced by C because the test of its meaning does not influence the output. In the case when C is true then the corresponding conjunct A_k is directly replaced by C.

The appendix illustrates the so formulated three-digit logic with two examples of real agents.

3.2. Defeating performance of BDI agent

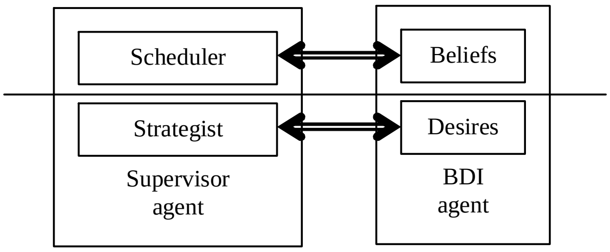

Fig. 5. Parity between components of supervisor and BDI agents

Introducing parities for (de)feasibility factors in cases of plans, strategies and tactics is a necessary sine qua non for research of defeating plans, strategies and tactics.

Another simplification is the ascertainment that inside the BDI agents the *technological necessity* factor falls off. It is replaced by the *technological indifference* factor (see item 3.1.1) respectively for the functional units beliefs-FUN, desires-FUN and intentions-FUN (the scheduler, the strategist and the tactician of the agent) because they are inherent by definition basic components of this type of agents. This problem falls off also for the action selector because defeating agent's performance proceeds in a direction opposite to its normal operation (see Fig. 4) so the internal for the agent defeating starts namely from the action selector[1].

Parities for (de)feasibility factors inside the agents can be obtained if we find the quantities with characteristic of *resources* respectively in beliefs-FUN, desires-FUN and intentions-FUN. The possible starting point with this regard can be the concept that the input to every of these FUN quantities determine the state of the resources in different moments of the functional unit. So we have a definite type of function

[1] Backtracking is advantegous to da cappo defeating (i.e. starting from beliefs-FUN) in the sense that it is time-saving. In fact the information stream from beliefs-FUN up to the action selector is cut off starting with defeating the (selected) action itself. If defeating starts from beliefs-FUN then there is some minimal time interval for spreading the information stream from beliefs-FUN up to the action selector included.

transforming the input data related to the specific problem of the functional unit into a distribution of the resources:

$R_{\text{FUN}} := f_{\text{FUN}}(\text{input}_{\text{FUN}})$, where:

FUN – respectively beliefs-FUN, desires-FUN, intentions-FUN or the action selector;

R_{FUN} – output from function f_{FUN} in the form of data for the distribution of the resources in FUN;

f_{FUN} – function of mapping the input data into R_{FUN};

$\text{input}_{\text{FUN}}$ – input data for FUN determining R_{FUN}.

The task scheduler is closest to the performance of such transformer in software products. If the usage of data from such scheduler is not convenient then it is relevant to make its approximation (similarity) or programly to realize the operation of this function; this realization is of principal importance for embedding the defeasible-inference mechanism inside the agent.

Note. The authors distinguish the already defined function (*severely individual* for every one of the three functional units beliefs-FUN, desires-FUN and intentions-FUN) from the performance of the agent's scheduler – beliefs-FUN. In the first case of the defined function the question is about the *resources allocation* in every one of the three functional units while in the case of the agent's scheduler beliefs-FUN the problem is about the *planning activities* of the agent itself.

A principle enhancement to avoid such function with the purpose of defeasible inference is using data structures that define the resources for the specific functional unit. In such approaches the quantity characterizing the output from the given functional unit is replaced by the quantity following immediately after it (by the next element in the queue, the stack, etc.).

A common feature of defeasible inference in every one of these four functional units is appointing the lowest possible values for the respective subsets of factors for the functional unit (see Table 2).

Table 2. Concretization of factors for defeasible inference in agents with BDI architecture

Factor	Beliefs-FUN	Desires-FUN	Intentions-FUN	Action selector
Technological necessity	Determined by the backtracking sequence for defeating FUN performance			
Probabilities	Probabilities for realizing plans	Probabilities for realizing strategies	Probabilities for realizing tactics	Probabilities for realizing actions
Priorities	Priorities of plans	Priorities of strategies	Priorities of tactics	Priorities of actions
Indicators (technological)	Technological indicators of plans	Technological indicators of strategies	Technological indicators of tactics	Technological indicators of actions
Resources	Function determining FUN resources or actualizing FUN data structures			

Note. The notions of resources and technological indicators include the minimal possible descriptions respectively of the resources and also of the technological indicators which descriptions guarantee the effective operation of the specific functional unit.

This appointing is followed by the replacement in the data structures of the defeated quantity by the quantity next after it in the data structure; it is determined by the type of the data structure and it is not directly related to the new action (goal) of the agent (in a sense the data structures are 'cleared' or *initialized* for the new goal).

4. Conclusions

The paper introduces the concept of defeasible inference for BDI intelligent agents based on inadequacy of the research up to now. The investigation starts with a model possessing features of BDI agent but with no cooperation with other agents. A taxonomy of concepts is presented: factors, conditions and criteria for social behavior of agents together with logical formalism for defeating rules of behavior; this formalism is illustrated by two intelligent agents. The research also includes peculiarities of collective behavior of BDI agents with probable defeating rules of the behavior.

References

1. G o r o d e t s k i,V., I. K o t e n k o, O. K a r s a e v. Framework for ontology-based represntation of distributed knowledge in multiagent netork security system. – In: Proc. of the 4th World Multi-conference on Systems, Cybernetics and Informatics (SCI-2000), V.III: "Virtual Engineering and Emergent Computing", Orlando, USA, July 2000, 52-58.
2. G o r o d e t s k i, V. Basic ideas of agent behaviour coordination and auction-based scheduling model. – In: In Proc. Of the I Int. Workshop "Distributed Artificial Intelligence and Multi-agent Systems", St. Petersburg, 282-291.
3. D o y l e, J., M. W e l l m a n. Preferential semantics for goals. – In: Proc. of AAAI-91, Anaheim, 1991, 698-703.
4. R a m s e y, F. Truth and probability. – In: R. B. Braithwaite (Ed.). The Foundations of Mathematics and Other Logical Essays, London, Kegan Paul, 1931, 156-198.
5. K o j n o v, S., V. S g u r e v, V. J o t s o v. The design process of intelligent agents as parallel elaboration. – In: Proc. of 2004 2nd Int. IEEE Conf. Intelligent Systems. V. I. Ronald R. Yager and Vassil S. Sgurev (Eds.), Varna, Bulgaria, June 22-24, 2004, St.st. Constantine and Helena resort, 2004, 267-272.
6. K o j n o v, S. Fast industrial intelligent agents. – In: Parallel Elaboration and Preprocessing, IIT Working Papers, Bulgarian Academy of Sciences, IIT/WP-182, November, 2004. 11 p.
7. Developing Enterprise Systems with Intelligent Agent Technology, by Faramarz Farhoodi and Peter Fingar, © 1997, Farhoodi and Fingar.
   home1.gte.net/pfingar/docmag_part2.htm
8. S a u n d e r s, R. A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy, Department of Architectural and Design Science, Faculty of Architecture, University of Sydney, February 2002.
   www.arch.usyd.edu.au/~rob/study/publications/thesis/html/thesis.html
9. **www.wiley.co.uk/college/turban/glossary.html**
10. Personoids' Organizations Framework: An Approach to Autonomous Software Systems Architecture Based on Intelligent Agents Organization (a preliminary study). Adam M. Gadomski, ENEA, Casaccia Research Center, Rome, Italy, April 1998.
    erg4146.casaccia.enea.it/Personoid-MAS2.pdf
11. Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence. Gerhard Weiss, Ed. Cambridge, Massachusetts, London, England, The MIT Press, 2001.
12. D u n g, P., P. M o n c a r e l l a. Production systems with negation as failure. – IEEE, Trans/ Knowledge and Data Engineering, vol. 14, April 2002, No 2, 336-352.
13. R e s c h e r, N. Many Valued Logics. Oxford, Basil Blackwell, 1979.
14. Я б л о н с к и й, С. Введение в дискретную математику. Москва, Высшая школа, 2001.

15. G l a d u n. Process of new knowledge formation. Sofia, Pedagog 6, 1994.
16. M a r k o v, Z. Inductive Machine Learning Methods. TEMPUS JEN 1497. Sofia, Softex, 1996.
17. J o t s o v, V. Knowledge discovery and data mining in number theory: some models and proofs. – In: Proc. Methods and Algorithms for Distributed Information Systems Design. Institute for Information Transmission Problems of RAS, 1997, 197-218.
19. K o w a l s k i, K. Logic for Problem Solving. Amsterdam, North-Holland Inc., 1979.
20. R a o, A. AgentSpeak(L): BDI agents speak out in a logical computable language. – Technical Note 64, Australian Artificial Intelligence Institute, Feb. 1996. Also appeared in Agents Breaking Away (LNAI 1038, 42-55).
21. J o t s o v, V., V. S g u r e v, A. T i m o f e e v. Applications of nonclassical logic methods for purposes of knowledge discovery and data mining. – International Journal Information Theories & Applications, **11**, 1-10.
22. S g u r e v, V., V. J o t s o v. Some characteristic features of the application of three-valued logical schemes in intelligent systems. – In: Proc. First International IEEE Symp. Intelligent Systems (T. Samad and V. Sgurev, Eds.). Vol. I. Varna, Bulgaria, September 10-12, 2002, 171-176.
23. S g u r e v, V., V. J o t s o v. Some defeasible inference methods by using network flows. – J. Problems of Engineering Cybernetics and Robotics, 2001, No 51, 110-116.

Аннулирование интеллектуальных агентов типа BDI

Стефан Койнов, Владимир Йоцов, Васил Сгурев

Институт информационных технологий, 1113 София

(Р е з ю м е)

В работе вводится принцип аннулирования агентов типа BDI, основанный на неполноте исследований до настоящего момента. Изложение начинается с представления модели с признаками агента типа BDI, который, однако, агентом некооперативного типа. Вводится система принципов: факторов, условий и критериев социального поведения агентов совместно с логическим формализмом аннулирования правил поведения. Этот формализм иллюстрирован примерами двух интеллигентных агентов. Исследование также включает особенности коллективного поведения агентов типа BDI с возможным аннулированием правил поведения.

Appendix: Logical formulas of two agents

Agent A_{12} (*intelligent*, Raw data preprocessing agent)

$A_{12}: S_1 \wedge (\mathit{prefix\,of}\ S_{2,3} \wedge S_2 \wedge S_3 \wedge \mathit{tail\ of}\ S_{2,3}) \wedge S_4 \wedge S_5 \wedge S_6 \wedge S_7 \wedge S_8 \wedge S_9$;

$S_1: S_{11} \wedge S_{12} \wedge S_{13} \wedge S_{14}$; $S_{1i}: \left\langle \mathit{result\ of}\ S_{1i} \leftarrow \mathit{verification\ of}\ S_{1i} \right\rangle$; where

verification of S_{11}: *measured data* $\leftarrow$ *efficient data dauges* $\leftarrow$ *data within the data – dauge range* (groups 1, 2),

verification of S_{12}: (*measured emission data* $\leftarrow$ *efficient CEM* $\leftarrow$ *emission data concentration within the CEM range*)$\wedge$(*calculated softsensing data* $\leftarrow$ *efficient DCS* $\leftarrow$ *DCS operability conditions*) (groups 3, 4),
verification of S_{13}: *done* $\leftarrow$ (*loading* > 70%) $\vee$ (*loading* < 70%)$\leftarrow$ *conditions for verification of emission sources*,
verification of S_{14}: *done* $\leftarrow$ (*presentcleaners*) $\vee$ (*missingcleaners*)$\leftarrow$ *conditions for cleaners – presence verification*;
S_{23}: *they participate in the formula done* $\leftarrow$ (*system-or-object adequacy to regulations*) $\vee$ (*system-or-object inadequacy to regulations*) $\leftarrow S_1$, where *system-or-object inadequacy to regulations*: $S_2 \wedge S_3$ and
S_2: *writing additional data*, S_3: *message for inadequacy*;

S_4: *done* $\leftarrow \bigwedge_j$ ((*necessarynumber of discrete values*)$\vee$ $\neg$(*necessarynumber of discrete values*))$\wedge \bigwedge_k$ ((*'yes' evaluated softsensing par k*) $\vee$ (*'no' evaluated softsensing par k*);

S_5: *done* $\leftarrow \bigwedge_n$ ((*adequate par.n*)$\vee$ (*inadequate par.n*)) (reliability of data bounds),

S_6: *done* $\leftarrow$ (*tech. param.and status – signals consistency*)$\vee$ (*tech. param.and status – signals inconsistency*) where
ech. param.and status – signals inconsistency: writing additional status data;
S_7: *recalculating data based on a given* O_2 *concentration*;
S_8: *communication with* A_{13} *for sending processed data*;
S_9: (*check of the 24-hour period for a total of 12 30 minute intervals*)$\wedge$(*direct the rest of the data towards* A_{43}).

<u>Agent A_{41} (*primitive*, Agent for standard processing)</u>

A_{41}: $S_1 \wedge S_2 \wedge S_3 \wedge S_4 \wedge S_5 \wedge S_6 \wedge S_7$;

S_1: $\bigwedge_i$ (*determine* 30 *minute average value for param. i*);

S_2: *form* 20 *standart classes for* 24 *hour period* = *f*(*deviation from NDE, registration frequency*);
S_3: *average values* = *f(*suitable 30 minute intervals);
S_4: *form monthly classes*;
S_5: *calculate monthly – average values*;
S_6: *calculate* 364 *combinations of* 48 *hour intervals for evaluation* where

done $\leftarrow \bigwedge_j$ ((*monthly j-th average* > NDE)$\vee$(*monthly j-th average* < NDE))$\wedge \bigwedge_k$ ((97% *of the* 30 *minute average values for* SO_2 *and dust exceed* 110% *of* NDE) $\vee$ (97% *of the* 30 *minute average values for* SO_2 *and dust do* *not exceed* 110% *of* NDE)$\wedge \bigwedge_m$ ((95% *of the* 30 *minute average values for* NO_x *and dust exceed* 110% *of* NDE) $\vee$ (95% *of the* 30 *minute average values for* NO_x *and dust do not exceed* 110% *of* NDE)$\wedge \bigwedge_n$ ((*measured n-th value from other kinds exceeds* NDE) $\vee$ (*measured n-th value from other kinds does not exceed* NDE)) and

$S_6 = \bigwedge_j$ (*monthly j-th average* < NDE)$\wedge \bigwedge_k$ (97% *of the* 30 *minute average values for* SO_2 *and dust do not exceed* 110% *of* NDE)$\wedge \bigwedge_m$ (95% *of the* 30 *minute average values for* NO_x *and dust do not exceed* 110% *of* NDE)$\wedge \bigwedge_n$ (*measured n-th value from other kinds does not exceed* NDE).