The actions of the agents in upper production control

Ofelia Vasilescu

National Institute for Informatics
Bvd. Averescu 8-10
71316 Bucharest
ROMANIA
e-mail: vofelia@u3.ici.ro

Abstract: In this paper we present the logical theory of goals and programs in a multi-agent system for upper production control. The achievement of an upper assortment is a complex decisional process into which, according to consumer requirement the producer interest must establish quick and optimal commercial, technical, esthetical and functional specifications as well as raw material option.

Key words: production control, cooperation, agent, actional situation, goal, ability, performance, skill, duration.

Ofelia Vasilescu is a Senior Researcher at the National Institute for Research and Development in Informatics-ICI. Bucharest. She received his MSc. in Computer Science from the Faculty of Automatic, section Computers, "Politehnica" University of Bucharest in 1978. She is a PhD student in Computer Science at the "Politehnica" University of Bucharest and PhD student in Logic at the University of Bucharest. Her research interests are: multi-agent systems, decision support systems, knowledge based systems, logic of action, fuzzy logic.

1. Introduction

The logical theory of goals and programs in this multi-agent system is to describe the use of the practical directives. The practical directive are rules and principles of actions attempted by an agent in a actional situation to achieve the objective or the goal had in view.[Kotarbinsky,1976] Within the framework of Prolog rules for upper production we capture several relevant action list concepts such as: ability, accessible goal, effective or successful action, autonomous achievable goal, heterogeneous achievable goal and others. Based on these concepts we present the scheduling ontology focused on the primitive entities for the product plans [Vasilescu, 2000].

Now in this paper, we present the alternative plans for shoe upper production and we describe the actions performed by the knowledge agent for the first phases: soaking-liming.

This article is an alternative of the finite automata and associated formal languages of Popa[1991, 1996] and a continuation of the formal logical theory of human actions developed by Popa in the last two decades.

We will present the logical theory of goals and programs based on the first order predicate logic. This approach was inspired by Lin et al.[1994, 1995] where first order logic was used to describe the effect of actions in the situation calculus.

2. The alternative plans for shoe upper production

Assortment production programming is a component of current industrial practice and means raw material selection as well as working variant establishing (phases and technological operation) in order to confirm the user requirements in the conditions of "0" rejections delivery. Because of his discreet type shoe upper production, the programming will be achieved in four phases, each one as a continuous process. Each phase is coordinated by a knowledge agent. The technological operations on each phase in the table from the figure 1, is presented.

PHASE 1 SOAKING-LIMINING	PHASE 2 MINERAL TANNAGE	PHASE 3 WET FINISHING (structuraly)	PHASE 4 SURFACE FINISHING
Soaking	Washing	**Wetting Filaments & fibres removal**	Surface degreasing
Degreasing	Degreasing	Degreasing	Impregnation
Disinfecting	Scudding	Acid detanning	Drum dyeing correction
Limining	**Deliming**	Chrome retanning	Grounding
Post- limining	**Bating**	**Neutralisation**	**Surface covering**
Washing	**Pickling**	Dyeing	**Dressing**
	Tanning	**Fatliquoring**	Pull up
	Basifying	Hidrofobisation	**Touch effect**
		Retanning	**Gloss effect**
		Humidity regulation	
		Cationic top	

Figure 1: The operations on the technological phases (after Bostaca et al.,1997)

The concatenation of the operation on technological phases presented in figure 1 can be described with graphs. The graph nodes S_i represent the upper states and the arcs a_i are associated with the operations.

Now, we present the soaking-liming phase which will be realised by the knowledge agent agk_l.

Soaking-Liming Phase is described by the graph from figure 2. The arcs of graph denotes the following operations:

a_1 = soaking
a_2 = *degreasing*
a_3 = limining
a_4= *disinfecting*
a_5 = limining
a_6 = limining
a_7 = *post-limining*
a_8 = washing
a_9 = washing

The operations can be obligatory as soaking , liming and washing or optional as degreasing, disinfecting and post-limining. The obligatory operations are automatically instead in replaced with data facts as obligatory operations and the optional operations are inserted based on system user answer.

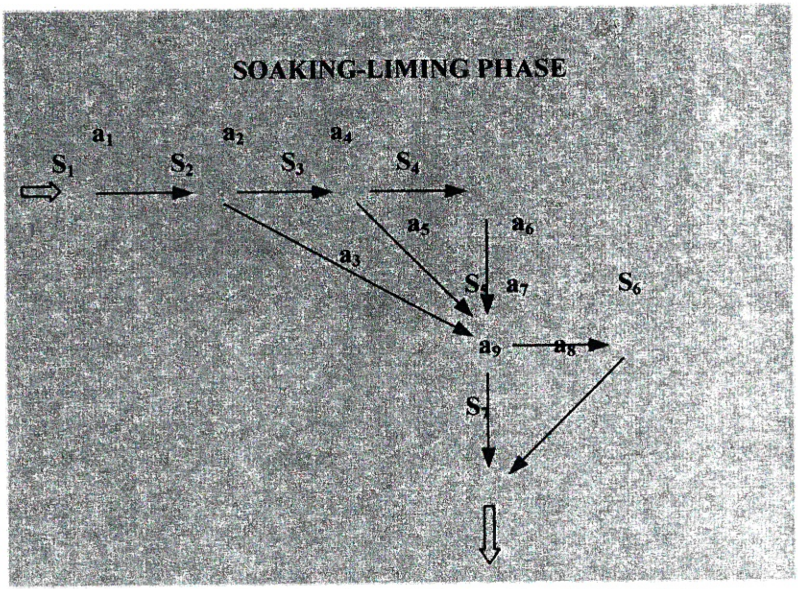

Figure 2: The operations of the soaking-liming phase

The first node S_1 represents the initial state, i.e. the brute state. The next nodes S_2 - S_7, indicate the result of a performed operation. The nodes S_1 - S_7 have the following semnification:

S_1 = upper_brute
S_2 = upper_ soaking
S_3 = upper_ degreasing
S_4 = upper_ disinfecting
S_5 = upper_ liming
S_6 = upper_ post-liming
S_7 = upper_gelatine

The set of operations handled to the achieve the goal, i.e. the demand of product. The goal is realised if the resource exists at the momemt of demand and the knowledge agent has the ability to perform the phase plan of product. The knowledge agent for this phase has six alternatives of actions:

(a1) $\{ a_1, a_2, a_4, a_6, a_7, a_8 \}$
(a2) $\{ a_1, a_2, a_4, a_6, a_9 \}$
(a3) $\{ a_1, a_2, a_5, a_7, a_8 \}$
(a4) $\{ a_1, a_2, a_5, a_9 \}$
(a5) $\{ a_1, a_3, a_7, a_8 \}$
(a6) $\{ a_1, a_3, a_9 \}$

Each alternative represents a plan for the demand of product. The knowledge agent selects the plan according to the product demand given by the coordination agent.

3. The actions of agents show that practical directives

We will describe the plan performed by the knowledge agent by means of the practical directives. A practical directive is a rule in which an agent exists in a situated action S_i at moment T_i and arrives in the goal state S_e at moment T_e after a total duration D_t and a total cost C_t if has ability to performe an action (an operation) N with resource R and the transition from S_i to S_e has a duration D and a cost C.

> fullfield(action(Ag_k,N,S_i,S_e,D_t,C_t),T_i, T_e):- holds(state(Ag_k,S_i,D_s,C_s), T_i), goal(Ag_k,S_i,S_e), holds(resource(Ag_k,P,Q,N), T_i), operation(N,S_i,S_e,D,C), ability(Ag_k,N).

We now define the syntax of language. The alphabet, as usual for first order predicate logic, consists of a set C of constants, a set V of variables, a set F of function symbols, a set P of predicate symbols, a set of connective symbols and a set of punctuation symbols. The Universe of Discourse consists of different kinds of objects including agent, time points, actions and states. In order to make it possible to distinguish between terms denoting these different kinds of objects, the terms are typed. We introduce the types Ag for agents, T for time points, A for actions, D for action durations, C for action costs and S for states. We also introduce the type AS for the contents of speech acts, which may be actions, states, or combinations of these; A and S are subtypes of AS. In order to make it possible to construct different actions and states, some special set of function symbols are defined:

- a set $I = \{\mathrm{dir}_c\} \subset F$ of illocutionary points;
- a set $A = \{a_1, a_2, \ldots, a_{48}\} \subset F$ of actions;
- a set $S = \{s_1, s_2, \ldots, s_{34}\} \subset F$ of states;
- a set $D = \{\mathrm{O}, \mathrm{Pe}, \mathrm{Fo}\} \subset F$ of deontic operators;

These function symbols are typed as follows:

$\wedge : A \times A \rightarrow A$
$\vee : A \times A \rightarrow A$
$dir : Ag \times Ag \times A \times T \rightarrow A$
$action : Ag \times A \times S \times S \times D \times C \rightarrow A$
$state : Ag \times S \times D \times C \rightarrow S$
$\mathrm{O} : Ag \times AS \times T \times T \rightarrow AS$
$\mathrm{Pe} : Ag \times AS \times T \times T \rightarrow AS$
$\mathrm{Fo} : Ag \times AS \times T \times T \rightarrow AS$

The predicate symbols in this language are $<$, $\leq$, $=$, *done* and *holds* with arity two and *fulfilled* with arity three. The predicate symbols *holds* , *done* and *fulfilled* are typed as follows:

$holds : S \times T$
$done : A \times T$
$fulfilled : A \times T \times T$

A practice directive will be a clause in Clausal Normal Form [Lloyd, 1987], that is an ordinary logic programming formula. We use Prolog notation and adopt the convention that constants are denoted with lower-case letters and variables with upper-case letters.

The function symbols introduced above shall be read as follows:

$dir\ (Ag_1, Ag_2, A, T)$- Ag_1 asks Ag_2 to fulfill the action A latest at T;

$state(Ag,S)$- The agent Ag is in the state S;

$action(Ag,N,S_1,S_2,D,C)$ -The agent Ag change from the state S_1 to S_2 through performed action N with a duration D and a cost C;

`O(Ag,AS,T1,T2)` –It is obligatory for `Ag` to fulfill `AS` between T_1 and T_2;

`Pe(Ag,AS,T1,T2)` –Agent `Ag` is permitted to fulfill `AS` between T_1 and T_2;

`Fo(Ag,AS,T1,T2)` –Agent `Ag` is forbidden to fulfill `AS` between T_1 and T_2;

The predicate built by *holds, done* and *fulfilled* shall be read as follows:

holds(S,T)– The state *S* holds at moment *T*;

done(A,T)– The action *A* has performed at moment *T*;

fulfilled(AS,T1,T2)– AS is performed between T_1 and T_2.

The predicate built by *holds* is used to show effect of the action *A* has performed between T_1 and T_2 when the state of upper is change from S_1 to S_2, at moment T_2, when action *A* has been performed.

```
A1. holds(state_upper(S2,D,C),T2)←
        fulfilld(action(Ag,N,S1,S2,D,C),T1,T2).
```

If the **upper** state holds in the state S_2 at moment *T* then the agent *Agk* moves from the state S_1 to S_2, at moment T_2, if the state S_2 is not the end state. If the state S_2 is the end state then the agent *Agk* moves to the state S_0 and can accept an other demand.

```
A2. holds(state(Agk,S2,D,C),T)←
        holds(state_upper(S2,D,C),T),
        state_end(Agk,Sf),
        S2 <> Sf .
A3. holds(state(Agk,s0,0,0),T)←
        holds(state_upper(S2,D,C),T),
        state_end(Agk,Sf),
        S2 = Sf ,
        assert(answer(agk1,D,C),answer_agk1)
    or
        state_init(Agk,s0).
```

The predicate built by *fulfilled* denotes execution of the action *A* between T_1 and T_2. Preconditions of the action *A* are:

- the agent Ag_k is in the state S_1, at the moment T_1,
- in the state S_1 is possible to perform the operation *N*, with the duration *Du* and the cost *Cs*,
- the agent Ag_k are ability to perform the operation *N*,
- the agent Ag_k are resource at the moment T_1 for the operation *N*.

```
A4. fulfilled(action(Agk,N,S1,S2,D,C),T1,T2)←
        holds(state(Agk,S1,Dt,Ct),T1),
        operation(N,S1,S2,Du,Cs),
        ability(agk1,N),
        holds(resource(Agk1,P,Q,N),T1),
        T2 = T1 + Du,
        D = Dt + Du,
        C = Ct + Cs.
```

Let us illustrate the execution of actions by the knowledge agent *agk1* for the soaking-liming technological phase. The initial state and the ability of agent are given by the following facts:

```
F1. state(agk1,s1).
    know(agk1,[soaking, degreasing, liming, disinfecting, post-liming, washing]).
    plan(agk1,[soaking, liming, post-liming, washing]).
    ability(agk1,[soaking, liming, post-liming, washing]).
    operation(soaking,s1,s2,2,100).
    operation(degreasing,s2,s3,0.5,25).
    operation(liming,s2,s5,1.5,100).
    operation(disinfecting,s3,s4,0.5,30).
    operation (liming,s3,s5,1.5,100).
```

operation(liming,s_4,s_5,1.5,100).
operation(post-liming,s_5,s_6,0.5,50).
operation(washing,s_6,s_7,0.5,50).
operation(washing,s_5,s_7,0.5,50).
resource(soaking-liming, soaking, detersinDBS).
resource(soaking-liming, liming,[lime_hydrated, sulphide_sodium, molasses])
resource (soaking-liming, post-liming, lime_hydrated,9%)
prescription_tech(soaking-liming, [detersinDBS,0.3%,lime_ hydrated,6%, sulphide_sodium,2.7%, molasses,9%])

We consider the following demand to process 800 kilograms of uppers for the assortment box in maxim 50 days from the moment of demand. The moment of demand is 1 and the initial state of upper is the brute state. The demand is illustrated by the facts:

F2. done(product_demand(box,800,50),1).
holds(state_upper(s_1,0,0)).
state_end(ag_{k1}, s_7).

From F2 and A2 follows the fact C1 that denotes the initial state of the agent *agk1*at the moment 1, in the state s_1, with the duration and the cost 0:

C1. holds(state(agk1,s_1,0,0),1)

The fact C1 starting the soaking-liming phase executed by the agent *agk1*. From fact C1 and the rule A4 we can derive the fact C2, as a result of execution of soaking operation between the states s_1 and s_2, at moment 1, with a duration of 2 days and a cost of $100.

C2. fulfilled(action(ag_{k1},soaking,s_1,s_2,2,100),1,3)

From C2 and A1 derives fact C3 therefore the upper state became s_2 after 2 days and with a cost of $100.

C3. holds(state_upper(s_2,2,100),3)

From C3 and A2 infers the fact C4 in which the agent ag_{k1} moves in the state s_2, at the moment 3 .

C4. holds(state(ag_{k1},s_2,2,100),3)

From C4 and the rule A4 derives the fact C5 following the next operation from plan of the agent ag_{k1}. The duration for both operations will be 3.5 days with a cost of $200.

C5. fulfilled (action(ag_{k1},cenuşarire,s_2,s_5,4.5,200),3,4.5).

From C5 and the rule A1 derives the fact C6 in which the upper state became s_5.

C6. holds(state_upper(s_5,4.5,200),4.5).

From C6 and the rule A2 follows the fact C7 therefore the agent *agk1* moves in the state s_5.

C7. holds(state(ag_{k1},s_5,4.5,200),4.5).

From C7 and the rule A4 we deduce the fact C8.

C8. fulfilled (action(ag_{k1},post_cenusarire,s_5,s_6,5,250),4.5,5).

Then from C8 and A1 follows:

C9. holds(state_upper(s_6,5,250),5).

Now from C9 and A2 we can derive that:

C10. holds(state(ag_{k1},s_6,5,250),5).

Then from C10 and A4 follows:

C11. fulfilled (action(ag_{k1},spalare,s_6,s_7,5.5,300),5,5.5).

Now from C11 and A1 we deduce the fact C12 in which the upper state is the final state s_7, while from C12 and A3 we infer C13 therefore the agent ag_{kl} moves in the initial state s_0, and will be able to perform other demands. The duration of demand was 4.5 days with a cost of $300. The answers of the knowledge agent ag_{kl} for the coordination agent will be memorized under the form of facts in Fact Data Base $answer\_ag_{kl}$.

C12. holds(state_upper(s_7,5.5,300),5.5)
C13. holds(state(ag_{k1},s_0,0,0),5.5).

In this way we present the actions performed by the agent ag_{kl} for a product demand asked by the coordination agent.

4. Conclusions

We presented the knowledge agent actions performed in a first order framework based on the practical directives. The practical directives are used to represent the alternative plans for shoe upper production and they describe the actions performed by the knowledge agent for the first phases: soaking-liming. For any phase of technological process, we consider an *initial state*, a *terminal state*, and a sequence of *actions or operations* by means of which transitions from one state to another take place, in a given interval of time. We presume that the action has a purpose or is guided by a goal asked by the coordination agent. The system offers to human decedent an optimum variant in programming technological processes and resources according to an expected option.

REFERENCES

1. BOSTACA G., O. VASILESCU BUȚA, N. BADEA, **Expert Systems for upper production programming optimization,**. the International Union of Leather Technologies and Chemists Societies Centenary Congress, 11-17 september, London, pp.703-708, 1997.
2. LIN, F., REITER, R., **Rules as Actions: A Situation Calculus Semantics for Logic Programs**, in J. Logic Programming, pp.679-710, vol. 19, 1994.
3. LIN, F., SHOHAM, Y., **Provably Correct Theories of Action**, in J. ACM, vol 42(2), pp.293-320, 1995.
4. POPA C., **Eilenberg Automata, Formal Languages and Semantics of Modal Logic**, Revue Roumaine de Philosophie, Tome 35, no.3-5, pp.163-171.
5. POPA C., **Praxiology, Logic of Action and Rationality of Human Activity,** in Praxiologies and the Philosopy of Economics, eds. J.Lee Auspitz, Wojciech,W. Gasparski, Marek K. Milici, Klemens Szaniawski, Transaction Publishers, New Brunswick, New Jersey, SUA, pp. 537-583.
6. USCHOLD, M., **Building Ontologies : Towards a Unified Methodology,** the 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems, Cambridge, 16-18 December, 1996.
7. SMITH, S., O. LASSILA, M. BECKER , **Configurable, mixed-initiative systems for planning and scheduling,** in Tate A. (Ed.), Advanced Planning Technology. Menlo Park: AAAI Press, 1996.
8. VASILESCU BUȚA, O., **Logica acțiunii și sistemele multi-agent**, GRAND al Academiei Române, 2000.
9. VASILESCU BUȚA, O., **Ontologia planificării producției**, RRIA, 10(4), 2000.
10. VASILESCU BUȚA, O., **Modelling the Communication Process between Agents,** In Studies in Informatics and Control, 9(4), 2000.