15th Triennial World Congress, Barcelona, Spain

SPECIFICATION AND SUPERVISORY CONTROL FOR MULTI-AGENT PRODUCT SYSTEMS

I. Romanovski [1] **and P.E. Caines**

Department of Electrical and Computer Engineering, McGill University
3480 University st., Montréal, QC, H3A 2A7, Canada
{peterc,yasha}@cim.mcgill.ca

Abstract: The prescription of sets of trajectories for controlled finite deterministic automaton G is formulated via the notion of the class of specifications denoted, both individually and collectively, by $SPEC$. Next, the formulation and (language) specification of structures for interacting automata are developed within the *Multi-Agent (MA) product* framework (Hubbard and Caines, 1999), and specifications are defined in terms of SPECs. Necessary and sufficient conditions for the synthesis of MA supervisors are given and an associated *MA product of specifications* is introduced; finally, illustrative examples for the results in the paper are provided.

Keywords: trajectory specification, supervisory control, automata, multi-agent systems, vector words, vector processes

1. INTRODUCTION

Systems in the areas of manufacturing, telecommunications, and transportations are often represented by networks of interacting objects, and in many cases specifications for such systems are naturally formulated in terms of transitions between system states. More specifically, such tasks may include visiting an ordered sequence of states (with possible constraints on visiting other system states) regardless of the event sequence by which this is achieved. For example, consider the operation of paying for merchandise in a shop. Regardless of the type of payment (credit card, debit card, check, etc.) it must be completed successfully by an authorization. Such design and control problems arise for the scalar systems represented by finite deterministic automata, as well as for vector (multi-agent) systems. For the latter, we use a formal theoretic framework of *Multi-Agent (MA) product* systems introduced in Hubbard and Caines (1999). Further development of the ideas for the analysis, control, and optimization of such systems is to be found in Romanovski and Caines (2001a,b). The results of this paper constitute a natural extension of the classical supervisory control results (see Kumar and Garg,(1995), among others) for scalar systems to the more general MA product system case. Illustrative examples for the results in the paper are provided in Sections 3 and 4.

2. SCALAR SPECIFICATIONS FOR FINITE AUTOMATA

Definition 1. A *specification* $(SPEC)$ for a given automaton G is a 4-tuple of subsets of X, namely,
$SPEC = \{X_I, X_T, X_{pc}, X_{bad}\}$, where
$X_{pc} \cap X_{bad} = \emptyset$. X_I is termed the set of initial states (of the $SPEC$), $X_T \subseteq X_m$ is termed the set of terminal states, X_{pc} is an *ordered* subset of X (possibly with repetitions) termed the set of ports of call and X_{bad} is termed the set of bad states. □

[1] Work supported by NSERC grant 1329-00

X_{pc} is the set of states which should be visited in order while X_{bad} is the set of states which must be avoided. Further, unless otherwise stated, X_I and X_T are singletons ($\{x_I\}$ and $\{x_T\}$ respectively).

The term *to drive a state x (of an automaton) to state y* means that there exists an input word of controllable and uncontrollable events a such that when the automaton is in the state x and accepts the word a the automaton terminates in state y, equivalently, y is reachable from x via an input sequence $a \in \Sigma^*$.

Definition 2. We say that an automaton $G = (X, \Sigma, \delta, x_o, X_m)$ satisfies the $SPEC = \{x_I, x_T, X_{pc}, X_{bad}\}$ if there exists a system trajectory t which satisfies:

(1) The initial automaton state x_o is driven along t to the state x_I without entering the set X_{pc}.
(2) t contains all the elements of X_{pc} in the given order.
(3) The trajectory t from x_o to the x_T does not meet the set of potentially bad states,

where a potentially bad state is a state in X_{bad}, or a state from which a bad state is reachable by a sequence of uncontrollable events. □

The solution to the problem of satisfying a specification SPEC for a given automaton can be divided into two steps: (a) eliminate all potentially bad states X_{pbad} (b) within the resulting set, $X - X_{pbad}$, establish the existence of a trajectory that visits all elements in X_{pc} in the order given by SPEC. Both steps were discussed and solutions were developed in Romanovski and Caines (2001a,b).

To simplify the notation, we include the initial and terminal state into X_{pc} as the first and last element respectively and represent a $SPEC$ as a pair $< X_{pc}, X_{bad} >$. Denote by $L_m(SPEC)_G$ the set of all trajectories that satisfy the above definition (a formal definition for $L_m(SPEC)_G$ can be found in Romanovski and Caines,
(2001b)). Evidently, $L_m(SPEC)_G$ is often not prefix closed. In fact, we have the following

Proposition 3. Let G be finite automaton for which $X_m = X$, and let $< X_{pc}, X_{bad} >$ be a SPEC for G, $L_m(SPEC)_G \neq \emptyset$. If $|X_{pc}| > 2$ then $L_m(SPEC)_G$ is not prefix closed. If $X_{pc} = \emptyset$ (i.e. the specification has only bad states), then $L_m(SPEC)_G$ is prefix closed. □

Definition 4. An automaton $G \downarrow_{SPEC} = (Y, \Sigma, \delta_1, Y_o, Y_m)$ is called a *restriction of G according to SPEC* if (i) $Y = X - X_{pbad}$, (ii) $\delta_1 = \delta \downarrow_Y$, where $\downarrow$ denotes the restriction operation of the domain of a partial function to the indicated set, and (iii) $Y_o = X_o - X_{pbad}$, $Y_m = X_m - X_{pbad}$. □

Proposition 5. Let G be finite automaton, $X_m = X$, and let $< \emptyset, X_{bad} >$ be a SPEC for G. $L_m(SPEC)_G$ is controllable w.r.t. G if and only if $X_{bad} = X_{pbad}$.

Proof. Let $L_m(SPEC)_G$ be controllable, and let $\delta^*(x_o, a) = x \notin X_{bad}$. Then for $a \in L_m(SPEC)_G$ and for any uncontrollable event u defined at state x we have that $au \in L_m(SPEC)_G$ and hence $\delta(x, u) \notin X_{bad}$. Thus, there is no uncontrollable event that leads from $x \notin X_{bad}$ to a state from X_{bad}. By definition $X_{bad} = X_{pbad}$.

Let $X_{bad} = X_{pbad}$. Then whenever $x \notin X_{bad}$ and an uncontrollable u is defined at x, $\delta(x, u) \notin X_{bad}$, or, in other words, for any $a \in L_m(SPEC)_G$ and $au \in L(G)$ we have that $au \in L_m(SPEC)_G$. □

Corollary 6. Let G be finite automaton, for which $X_m = X$. Let $< \emptyset, X_{bad} >$ be a SPEC for G, and assume $L_m(SPEC)_G \neq \emptyset$. There is Σ_U-enabling supervisor for $L_m(SPEC)_G$ if and only if $X_{bad} = X_{pbad}$. □

Corollary 7. In the setup of the previous proposition, $L(G) \downarrow_{SPEC}$ is the maximal controllable sublanguage w.r.t. G. □

3. SUPERVISION OF MA SYSTEMS

The standard interaction for the supervisor-system pair is that of the synchronous product (see Kumar and Garg, (1995), for example). An automaton $S = (Y, \Sigma, \delta_S, y_0, Y_m)$ representing the supervisor operates with the plant $G = (X, \Sigma = \Sigma_c \cup \Sigma_u, \delta, x_0, X_m)$, and the resulting language is the scalar synchronous product $L(S)||_s L(G)$ (see Kumar and Garg, 1995).

An alternative is to consider control of a system G with a supervisor S acting in unison, as an individual agent, leading to the combined evolution $L(S)||_{MA} L(G)$. In what follows it is assumed that all languages are prefix-closed, hence both the terms $L(S||_{MA}G)$ and $L(S)||_{MA}L(G)$ can be used equivalently (note the latter is only defined for prefix closed languages). This assumption extends to specification languages (e.g. K below). It is also assumed that the goal states X_m and Y_m are the entire states X and Y. This has the effect of simplifying the algebraic derivations by alleviating the need for a *non-marking* condition for the supervisor (as in Kumar and Garg, 1995) and isolating the controllability criteria.

The results regarding controllability of a language and the synthesis of synchronous product based supervisors apply almost directly for scalar specifications K when the MA product is used in lieu of the scalar synchronous product. For vector specifications, however, controllability is not enough for the synthesis of MA-supervisors. This is due to the fact that in MA-product we often cannot disable an isolated (disablable) event, but only the controllable components of a given event.

Let the plant model consists of the MA product of automata and let the components be

$$G_i = (X_i, \Sigma_{i_c} \bigcup \Sigma_{i_u}, \delta_i, x_{0_i}, X_{m_i}), i = 1, 2..., N$$

where Σ_{i_c} are the disable events and Σ_{i_u} are the undisablable events. It is assumed that (for the case $N = 2$),

$$\Sigma_{i_c} \bigcap \Sigma_{j_u} = \emptyset,\ \Sigma_{i_u} \bigcap \Sigma_{j_c} = \emptyset,$$

which forces the events of the form $[a \dots a]^T$ to be uncontrollable or controllable in both components. The Definitions 8 and 9 below are written for $N = 2$, but easily generalised for an arbitrary N.

Definition 8. **Multi-Agent Product (Supervisory Case)**

$$G_1 ||_{MA} G_2 =$$
$$= (X_1 \times X_2, \Sigma_C \bigcup \Sigma_U, \delta_{MA}, (x_{0_1}, x_{0_2}), X_{m_1} \times X_{m_2})$$

where,

$$\Sigma_U = \{ \begin{bmatrix} a \\ b \end{bmatrix} \mid\ a \in \Sigma_{1_u} \text{ and } b \in \Sigma_{2_u} \} \qquad (1)$$

$$\Sigma_C = \{ \begin{bmatrix} a \\ b \end{bmatrix} \mid\ a \in \Sigma_{1_c} \text{ or } b \in \Sigma_{2_c} \} \qquad (2)$$

$$\delta_{MA}(\begin{bmatrix} x \\ y \end{bmatrix}, \begin{bmatrix} a \\ b \end{bmatrix}) = \begin{bmatrix} \delta_1(x,a) \\ \delta_2(y,b) \end{bmatrix},\ if$$

$\delta_1(x,a)!\ \wedge\ \delta_2(y,b)!\ \ \wedge (a = b\ \vee$
$(\neg\delta_2(y,a)! \wedge \neg\delta_1(x,b)!)\ )$,
and undefined otherwise. The notation $\delta_1(x,a)!$ means that δ is defined at (x,a). □

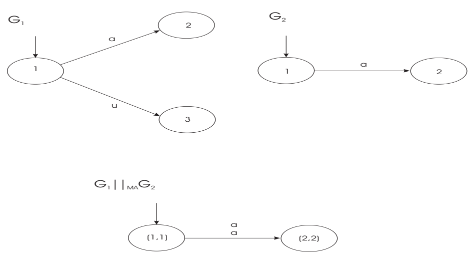

Fig.1: Mechanical disabling

Other constructions of product systems can be found in Kam at al., (1997), Hartmanis and Stearns, (1966) and Li and Wonham, (1993). It is easy to see that it is possible that some uncontrollable events defined for the automaton G_1 (or G_2) can be prevented by *synchronization*, as is shown on Fig.1. Moreover, even though the specification $K = \{a\} \subset L(G_1) = \{a, b\}$ is not controllable (we assume that b is uncontrollable), we have that $L(G||_{MA}S) = \{ \begin{bmatrix} a \\ a \end{bmatrix} \}$, since $\begin{bmatrix} b \\ a \end{bmatrix}$ cannot occur by the construction of the MA product.

In order to eliminate the prevention of uncontrollable event mechanically (i.e. by construction of the MA-product), we need to introduce a Σ_U*-enabling* MA-product. For the construction of supervisor S, it can be considered as an MA-analogy of scalar Σ_U*-enabling* (Kumar and Garg, 1995).

Assume that the MA-product of G_1 and G_2, namely, $G_1 ||_{MA} G_2 =$
$= (Z = Y \times X,\ \Sigma = \Sigma_C \bigcup \Sigma_U,\ \delta_{MA},\ z_0 = (y_0, x_0), Y_m \times X_m)$ is defined, and δ^*_{MA} is a natural extension of δ_{MA} on Σ^*, which is defined as

$$\begin{bmatrix} \Sigma_{G_1} \\ \Sigma_{G_2} \end{bmatrix}^* = \{ \begin{bmatrix} v \\ w \end{bmatrix} \mid v \in \Sigma^*_{G_1}, w \in \Sigma^*_{G_2}, |v| = |w| \}.$$

Definition 9. For any vector state $z \in Z$,

$$\mathcal{P}^{-1}_{G_1,z}(a) \overset{\text{def}}{=} \{ \begin{bmatrix} a \\ b \end{bmatrix} \in \Sigma | \delta_{MA}(z, \begin{bmatrix} a \\ b \end{bmatrix})! \},\ \ a \in \Sigma_1.$$

Similarly,

$$\mathcal{P}^{-1}_{G_2,z}(d) \overset{\text{def}}{=} \{ \begin{bmatrix} c \\ d \end{bmatrix} \in \Sigma | \delta_{MA}(z, \begin{bmatrix} c \\ d \end{bmatrix})! \},\ \ d \in \Sigma_2.$$

where $\Sigma_i = \Sigma_{U_{G_i}} \cup \Sigma_{C_{G_i}}$ defined for automata G_i, $i = 1, 2$ □

We generalise the notion of *component-wise projection* (see Hubbard and Caines, (1999), Romanovski and Caines (2001a))
$\mathtt{P}_i : (\Sigma_1 \times \ldots \times \Sigma_N)^* \longrightarrow \Sigma^*_i$ to $\mathtt{P}_{G_i}$ as follows:

Definition 10. Let the MA product $G_1 ||_{MA} G_2$ be defined. For any vector word
$s = [a_1, \dots, a_N, a_{N+1}, \dots, a_M]^T \in L(G_1 ||_{MA} G_2)$,
$[a_1, \dots, a_N]^T \in L(G_1), [a_{N+1}, \dots, a_M]^T \in L(G_2)$,

$$\mathtt{P}_{G_1}(s) \overset{\text{def}}{=} [a_1, \dots, a_N]^T,$$

and $\mathtt{P}_{G_2}$ is defined similarly. □

Definition 11. **Σ_U-enabling MA-product**.
$G_1 ||_{MA} G_2$ is Σ_U*-enabling* if the following condition holds:
whenever

$$z \in Z, s \in \Sigma^*, w_i \in \Sigma_{U_{G_i}},\ i = 1, 2$$

are such that $\delta^*_{MA}(z_0, s) = z$ and $\mathtt{P}_{G_i}(s) w_i \in L(G_i)$, then it is the case that

$$\mathcal{P}^{-1}_{G_i,z}(w_i) \neq \emptyset.$$

□

Note that the condition of the above definition implies $\emptyset \neq s\mathcal{P}^{-1}_{G,z}(w_i) \subseteq L(G_1) ||_{MA} L(G_2)$. In other words, if the vector state z is reachable from the initial state z_0 of $G_1 ||_{MA} G_2$ and some uncontrollable (vector or scalar) event $w_1 \in \Sigma_{U_{G_1}}$ (or $w_2 \in \Sigma_{U_{G_2}}$) is defined for a component of z that belongs to G_1 (or, respectively, to G_2), then there must be some event $v_2 \in \Sigma_{G_2}$ (respectively, $v_1 \in \Sigma_{G_1}$) such that $\delta_{MA}(z, [w_1, v_2]^T)$ is defined (respectively, $\delta_{MA}(z, [v_1, w_2]^T)$ is defined).

Definition 12. An MA supervisor S for G is called MA Σ_U-enabling if and only if $S||_{MA}G$ is a Σ_U-enabling MA product. □

Lemma 13. Let K and L be prefix-closed languages. Then,

$$K \subseteq L \implies$$
$$\implies K||_{MA}L = K||_{MA}K = \{ \begin{bmatrix} a \\ a \end{bmatrix} | a \in K \}.$$

□

For an MA supervisor we assume $\Sigma_S \subseteq \Sigma_G$.

Definition 14. An MA supervisor S for G is called *MA Σ_U-enabling* if and only if $S||_{MA}G$ is a Σ_U-enabling MA product. □

Definition 15. Let G be an MA-product system and K be a prefix-closed vector specification. K is called *MA controllable* (w.r.t. G) if and only if the the following conditions are true:

(1) K is controllable (i.e. $\overline{K}\Sigma_U \cap L(G) \subseteq \overline{K}$);
(2) $\forall a \in \Sigma_G, s \in K, x \in X\{\delta_G^*(x_0, s) = x \wedge sa \in L(G) \wedge sa \notin K\} \implies \exists a_i \in \Sigma_{i_c}, i = 1, \ldots, N$ $such\ that$ $([s\mathcal{P}_{G_i,x}^{-1}(a_i)] \cap K = \emptyset)$ □

We paraphrase the Condition 2 of the above definition as follows: if a controllable vector event a defined at a given vector state x, takes us out of the specification K, there must be some controllable component of a, say a_i, such that any vector event that is defined at x and has a_i as a component, takes us out of K. Note that if K is a scalar specification, the Condition 2 is trivially true.

Theorem 1. Let $K \subseteq L(G)$ be a regular (i.e. finitely generated) vector specification for an MA product system G.

(1) K admits a Σ_U-enabling MA supervisor S such that $\mathtt{P}_G(L(S)||_{MA}L(G)) = K$ if and only if K is MA controllable.
(2) If K is not MA controllable, then there exists a maximal (w.r.t. the inclusion partial order) specification $K_1 \subset K$ which is MA controllable w.r.t. G.

Proof.
Part 1. We construct an $S = (X_S, \Sigma = \Sigma_C \cup \Sigma_U, \delta^S, x_0)$ by the following rules:

(1) $X_S = \{[s](R_K) | s \in K\}$, where R_K is an equivalence relation induced by K according to the Myhill-Nerode construction (see Kumar and Garg (1995)). Since K is regular, R_K is of finite index.
(2) For any state $x \in X_S$ any vector event $a \in \Sigma$,

$$\delta^S(x, a) = [sa](R_K)$$

if and only if $sa \in K$. For each vector event a such that $sa \notin K$ we disable (i.e. do not define) any controllable a_i at each component x_i of $x \in X_S$ for which

$$s\mathcal{P}_{G_i,x}^{-1}(a_i) \cap K = \emptyset$$

Other words, we make $L(S) = K$. Since K is controllable, $S||_{MA}G$ is Σ_U-enabling MA product, so S is Σ_U-enabling . By Lemma 12 we have that $\mathtt{P}_G(L(S)||_{MA}L(G)) = K$.

Assume that such S exists. Then K is controllable since $S||_{MA}G$ is Σ_U-enabling MA product. Assume there exist a vector state $x \in X$, vector word $s \in K$, and an event $a \in \Sigma_G$ for which the Condition 2 of the theorem is not true. We cannot leave this event in S since then $L(S) \neq K$. On the other hand, by the disabling of any component of vector event a we disable some event that belongs to K since for any $i = 1, ..., N$

$$s\mathcal{P}_{G_i,x}^{-1}(a_i) \cap K \neq \emptyset$$

But, again, $L(S) \neq K$ and so $\mathtt{P}_G(L(S)||_{MA}L(G)) = L(S) \neq K$. Contradiction.

Part 2. Consider two cases:
Case 1. K is controllable but not MA controllable. In this case the algorithm for finding a K_1 is the following: we start with the initial state x_o. If all vector events defined at this state satisfy the Condition 2 of the definition of MA controllability, we move to all states that are directly accessible (i.e. by one transition) from x_o. Suppose at state x condition 2 is violated. We remove events from K as follows: For each $s \in K, a \in \Sigma$ such that $\delta^*(x_0, s) = x$, and $sa \notin K$, we find a component a_i such that the cardinality of the set $s\mathcal{P}_{G_i,x}^{-1}(a_i) \cap K$ is minimal and remove from K all elements of the type $s\mathcal{P}_{G_i,x}^{-1}(a_i)$; then move to the states which are still accessible by the elements in the reduced language K'. Continue the procedure until all states accessible from x_o satisfy Condition 2. Note that since K is controllable, the resulting set K_1 will also be controllable and satisfy the condition 2 by construction, so K_1 will be MA controllable.

Case 2. Let K be uncontrollable w.r.t. G. There is a procedure (see Kumar and Garg (1995), among others) for finding the maximal controllable sublanguage of K, denoted by K_1'. As it is shown in Romanovski and Caines (2001a), K_1', in general, does not satisfy the Condition 2. In order to obtain K_1, we apply to K_1' the algorithm described above. □

Lemma 16. Let the specifications K_1 and K_2 be controllable w.r.t. the automata G_1 and G_2 respectively. Then $K_1||_{MA}K_2$ is controllable w.r.t. $G_1||_{MA}G_2$.

Proof. Let $s = [s_1, s_2]^T \in L(K_1||_{MA}K_2) \cap L(G_1||_{MA}G_2)$, $[u, v]^T \in \Sigma_U$, and $[s_1, s_2]^T[u, v]^T \in L(G_1||_{MA}G_2)$. Then $s_1u \in L(G_1)$ and, since K_1 is controllable w.r.t. G_1, $s_1u \in K_1$. Similarly, $s_2u \in K_2$. As a result, $[s_1, s_2]^T[u, v]^T = [s_1u, s_2v]^T \in K_1||_{MA}K_2$. □

Proposition 17. Let the specifications K_1 and K_2 be MA controllable w.r.t. the automata G_1 and G_2 respectively. Then $K = K_1||_{MA}K_2$ is MA controllable w.r.t. $G = G_1||_{MA}G_2$.

Proof. Due to the above lemma, it is enough to prove that the Condition 2 of the definition of MA controllability is true. Assume that for some vector state x of MA product $G_1||_{MA}G_2$, some vector word s such that $\delta^*_{MA}(x_0, s) = x$ and $s \in K_1||_{MA}K_2$, and some controllable vector event a defined at x we have that $sa \in L(G_1||_{MA}G_2)$, but $sa \notin K_1||_{MA}K_2$. Denote $x = [x_1, x_2]^T$, where $x_1 \in X_1, x_2 \in X_2$, $\mathtt{P}_{G_1}(sa) = s_1a_1$, $\mathtt{P}_{G_2}(sa) = s_2a_2$. Then either $s_1a_1 \notin K_1$ or $s_2a_2 \notin K_2$, or both. Since K_1 is MA controllable w.r.t. G_1, we have that there exists a controllable component a_{1_i} of a_1 such that $s_1\mathcal{P}^{-1}_{G_1,x_1}(a_{1_i}) \cap K_1 = \emptyset$. But that implies $s\mathcal{P}^{-1}_{G,x}(a_{1_i}) \cap K = \emptyset$. In the second case (as well as in the third) we get the same result by MA controllability of K_2. Thus, $K = K_1||_{MA}K_2$ is MA controllable w.r.t.$G = G_1||_{MA}G_2$. □

Let K_1 be the specification for G_1, K_2 be the specification for G_2. Naturally, we assume that $K_1 \subseteq L(G_1)$, $K_2 \subseteq L(G_2)$, and we consider the MA product $K_1||_{MA}K_2$ w.r.t. $G_1||_{MA}G_2$, which is formally defined as follows:

Definition 18. The MA specification $K_1||_{MA}K_2$ for the MA product $G_1||_{MA}G_2$ is given by
$(K_1||_{MA}K_2) \cap L(G_1||_{MA}G_2)$. □

Proposition 19. Let the specifications K_1 and K_2 be MA controllable w.r.t. the automata G_1 and G_2 respectively by the MA supervisors S_1 and S_2 respectively. If $G_1||_{MA}G_2$ is a Σ_U-enabling MA product, then the MA specification $K_1||_{MA}K_2$ has an MA supervisor $S_1||_{MA}S_2$ w.r.t. $G_1||_{MA}G_2$.

Proof. Denote $G = G_1||_{MA}G_2$, $K = K_1||_{MA}K_2$, $S = S_1||_{MA}S_2$. We have that $L(S) \subseteq L(G)$ and since G is Σ_U-enabling, $S||_{MA}G$ is Σ_U-enabling. Moreover, we have that $\mathtt{P}_{G_1}(S||_{MA}G) = K_1$,
$\mathtt{P}_{G_2}(S||_{MA}G) = K_2$, that gives $\mathtt{P}_G(S||_{MA}G) = K$.

Example 1. Consider the vector specification $K_1 = \{[a, c]^T, [b, d]^T\}$ for $G_1||_{MA}G_2$, given on Fig.2. K_1 fails to satisfy the condition 2 of the theorem, and it is impossible to disable any component without losing some element of K. Consider the additional specification for G_2, $K_2 = \{c\}$. Then vector specification $K_1||_{MA}K_2 = \{[a, c, c]^T\}$ is MA controllable w.r.t. $G_1||_{MA}G_2||_{MA}G_2$. □

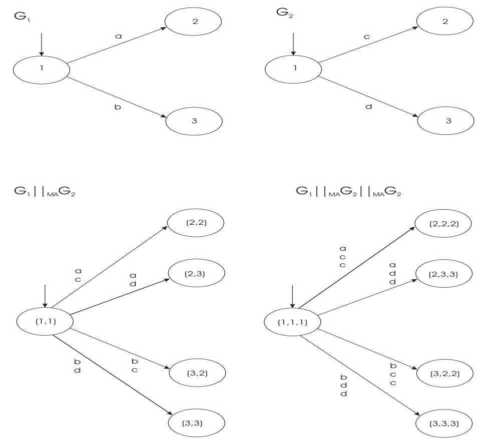

Fig.2: Examples of MA Products

4. VECTOR SPECS FOR MA PRODUCT SYSTEMS

There are two ways to analyse the specifications for MA product systems. Since the MA product is a finite deterministic automaton, we can construct specifications directly for such automaton and directly apply the results from Romanovski and Caines (2001b). The other approach is to give a specification for an MA system via specifications for its component systems.

Example 2. Consider the complex system of interaction of customer and sales department in the small shop. It is clear that whenever the customer has successfully paid for the item, we must remove it from the shelf. The specification is to keep shelfs non-empty when a customer is in the shop.

The behaviour of the Customer (G1) is represented in Fig.3. Here, the events out, $full\ stock$ (i.e. the demand of a customer to show all items available), $activate$, $complete$, $incomplete$ are not controllable, and the other events, $stay$, $try\ again$ and $refuse$ are controllable.

The scheme for a “shop” includes two automata:
“Counter” (G2) which represents a mechanism for paying and “Shelf” (G3), which carries information about items available. General automata for “Counter” and “Shelf” are given in Fig.3 also. We denote by $WAIT_{G2}$ the set of events $\{stay,\ out,\ full\ stock\}$ defined at the state $Idle$ of the automata $G2$. The events $complete_i, (i = 1 \ldots 3)$, $full\ stock_j, (j = 1, 2$ and $refill$ are controllable.

To be specific, we assume here that the capacity of the shelf is 3 items, and allow 3 attempts to complete the procedure of paying before the final refusal. It is clear that a similar automaton can be constructed for an arbitrary number of attempts and an arbitrary shelf capacity.

We represent the behaviour of the whole system as the MA product of $G1$, $G2$, and $G3$, denoted by G.

Evidently, our specification can be represented by the following SPEC: $< \emptyset, \{[Enter,\ *,\ 0]^T\} >$, where * is any state of the automaton G2. According to the Proposition 3, $L_m(SPEC)_G$ is prefix closed. Denote $K = L_m(SPEC)_G$. Further, $X_{bad} = X_{pbad}$ since we can get to the state *Enter* of the automata G1 only by controllable event *stay*, so K is controllable by the Proposition 5. We also note that K is MA controllable since the only vector event that takes us out of K is $(stay,\ stay,\ stay)^T$ defined at the state $[Get,\ Idle,\ 0]^T$.

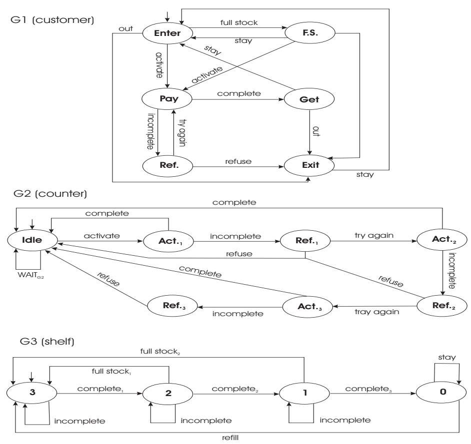

Fig.3: Automata for the MA product "Shop"

Now, the MA supervisor for the given specification K exists by Theorem 3.1. and can be constructed by disabling the the component $stay$ at the state $[Get,\ Idle,\ 0]^T$. Naturally, we assume that we remove an item from the shelf if and only if the procedure of payment is completed successfully, i.e. we enable the component $complete_k$, $k = 1, 2, 3$, only at the vector states $[Pay,\ Act._i,\ J]^T$, where $i = 1, 2, 3$, $J = 3, 2, 1$ and also, we try to put an additional items to the shelf only if there is a demand, so we enable the component $full\ stock_k$, $k = 1, 2$ only at the vector states $[Enter,\ Idle, J]^T$, $J = 1, 2, 3$. Note that the enabling of the event $full\ stock_k$ does not guarantee the return to the state 3 of the automaton $G3$, since the vector event $(full\ stock,\ full\ stock,\ incomplete)^T$ is defined at the vector state $[Enter,\ Idle,\ J]^T$, $J = 2, 1$, but this does not contradict our specification. □

5. REFERENCES

Hubbard, P. and P.E.Caines. (1999). Initial Investigations of Hierarchical Supervisory Control for Multi-Agent Systems. *Proceedings of the 38th IEEE CDC*, Phoenix, AZ, **3**, pp.2218-2223.

Hartmanis, J. and R.E.Stearns. (1966). *Algebraic Structure Theory of Sequential Machines*. Prentice Hall, Englewood Cliffs.

Kumar, R. and V.K.Garg. (1995). *Modeling and Control of Logical Discrete Event Systems*. Kluwer Academic Publishers, London.

Kam, T., T.Villa, R.Brayton, A.Sangiovanni-Vincentelli. (1997). *Synthesis of Finite State Machines: Functional Optimization* Kluwer Academic Publishers, London.

Y.Li and W.M.Wonham. (1993). Control of Vector Discrete-Event Systems I Base Model. *IEEE Transactions on Automatic Control* **38**, no.8, pp.1214-1227.

Romanovski, I. and P.E.Caines. (2001a). Analysis and synthesis of vector specifications for multi-agent systems. *In preparation*.

Romanovski, I. and P.E.Caines. (2001b). On Vector Trajectory Specifications for Multi-Agent Product Systems. *Proceedings of the 40th IEEE CDC*, Orlando, FL, pp.2333-2334.

Wonham, W.M. and P.J.Ramadge. (1987). On the Supremal Controllable Sublanguage of a Given Language. *SIAM J. Control Optimization* **25** 637-659.