APPROCHE DE SYNTHESE EN LIGNE BASÉE SUR UNE MODELISATION STRUCTURÉE DE LA PARTIE OPERATIVE
ON-LINE SYNTHESIS APPROACH BASED ON A STRUCTURED PLANT MODELLING

A. Philippot, A. Tajer, F. Gellot, V. Carré-Ménétrier

Centre de Recherche en STIC (CReSTIC), Moulin de la Housse B.P. 1039, 51687 REIMS Cedex 2, FRANCE. Tel: +33.3.26.91.32.26, Fax: +33.3.26.91.31.06
{alexandre.philippot, abdelouahed.tajer, francois.gellot, veronique.carre}@univ-reims.fr

Résumé: Les travaux présentés dans cet article reposent sur une approche de synthèse de commande optimale basée sur la spécification de la commande en grafcet et sur les modèles de la partie opérative et des contraintes à respecter par le système. Pour améliorer la praticabilité de l’approche sur des systèmes complexes et réels, nous proposons une méthodologie de modélisation de la partie opérative ainsi qu’une approche en ligne de la synthèse de la commande. Ces deux points sont formalisées dans l’article et traités à travers un exemple.

Abstract: This paper is based on an approach of synthesis and optimal control. For a given Grafcet, a plant and constraints models, this method must respect the process evolution. We developed two improvements of this approach to be used on complex real systems. The new results obtain concern the plant modelling and the on-line synthesis. Formal algorithms and an example are presented in the paper.

Keywords: Automata, Grafcet, Control synthesis, Plant modelling, Supervisory control theory, On-line synthesis.

1. INTRODUCTION

The objective of our work is to generate the required correct controller implementation for a given Grafcet (IEC 60848, 2002). This controller must to respect a number of safety and liveness constraints related to the modelled plant. The method is based on the use of a supervisor and a controller. The supervisor enforces the given safety constraints, whereas the controller directs the system to accomplish a specific set of tasks. An intuitive presentation of the different steps involved is given below, the formal algorithms can be found in (Carré-Ménétrier, et al., 1999).

The first step consists of modelling the plant behaviour, the safety and liveness constraints using automata. The specification of the control behaviour is described by Grafcet. The second step is used to generate the optimal controller by an intersection. This step is realised between the controller, the plant and constraints models (Carré-Ménétrier, et al., 1999). The maximum reactive non-blocking execution controller is then generated.

The last step consists of implementing the correct controller by taking into account the constraints. The proposed implementation architecture executes the correct controller and correspondingly updates the state of Grafcet. This step allow to visualise the execution of the control model and to highlight the corrections induced by the synthesis framework.

We showed the applicability of this method on several examples (Carré-Ménétrier, et al., 1999), (Philippot, et al., 2003). However, to be practicable on complex real systems, this approach encounters two problems. The first is lied in the plant modelling which is realised without methodology. The second in computational complexity of the oof-line synthesis. The aspects related to the plant modelling are presented around an example (fig. 1) in Section 2 by an adaptation of Chandra modelisation (Chandra and Kumar, 2001). Section 3 is dedicated to a particular synchronisation algorithm which proposes an off-line synthesis approach. The contribution of the present paper is the establishment of the on-line synthesis formalization in Section 4.
2. METHOD OF PLANT MODELLING

Our previous works showed that some knowledge of plant behaviour is required to be able to identify and to treat deadlocks. It is necessary to make the image of the plant explicit, by connecting an appropriate abstract model for each of its elements.

Hence, a modular approach is required to extract a simple model for each of the elements of the plant. Such a model can be derived using automata that accept the control actions, and react by changing the logical values of Grafcet inputs. The behaviour of the plant is modelled by means of the spontaneous event generators of the supervisory control theory (Wonham and Ramadge, 1987). However, this requires the use of an adequate methodology for the modelling of the plant, which is proposed in this paper. This methodology can be considered as an adaptation of Chandra’s approach (Chandra and Kumar, 2001) to be concisent with ours. The activation and deactivation of Grafcet actions correspond to controllable events \( \Sigma_c \). This because of the possibility to prevent their occurrence by an appropriate conditioning of Grafcet actions. Uncontrollable events, \( \Sigma_u \), are initiated by the plant. These events (\( \Sigma_c \) and \( \Sigma_u \)), which cannot be disabled by control action, are associated with the rising and falling edges of Grafcet inputs. This interpretation can be considered as an adaptation of Balemi’s scheme (Balemi, et al., 1993) to the case of Grafcet.

The adapted methodology is based on occurrence rules and precedence relations. These rules define the interactions between controllable or uncontrollable events. The precedence relations specify the links between the uncontrollable events. The user define the rules and relations which are translated into automata compatible with our synthesis approach.

2.1. Occurrence Rules and Precedence Relations

To determine the occurrence rules, it is necessary : (i) to fix the initial conditions of the system from the inputs vector of the controller model, (ii) to determine all the events related to the element of plant, (iii) to define with rules, the influence of the activation and the deactivation of the controllable events on the uncontrollable events. Each rule is a "cause/consequence". The cause relates the controllable event to the consequence (uncontrollable event). For each occurrence rule with the same cause, the chronology between the consequent uncontrollable events are established with precedence relations.

For the pneumatic gripper example (fig. 1), the complete model consists of 3 automata. They describe respectively the horizontal movement, the vertical movement and the movement of aspiration. These models take account of the cylinders technology. For the horizontal movement, there are two controllable events \( GO\_RIGHT \) and \( GO\_LEFT \), and two uncontrollable events \( left \) and \( right \). In initial situation, the cylinder is on the left and no order is sent to the plant. To define the occurrence rules, the consequences of the sent

\[
\begin{array}{ccc}
\text{Initial Conditions} & \text{Occurrence rules} & \text{Precedence relations} \\
\hline
\text{GO\_RIGHT} & \text{GO\_RIGHT} \rightarrow \text{left} & \text{left precede \( ? \right\)} \\
\text{GO\_LEFT} & \text{GO\_LEFT} \rightarrow \text{right} & \text{right precede \( ? \left\)} \\
\text{right} & \text{GO\_RIGHT} \rightarrow \text{right} & \text{nothing} \\
\text{left} & \text{GO\_LEFT} \rightarrow \text{left} & \text{nothing}
\end{array}
\]

Table 1. Parameters of the horizontal movement

2.2. Plant automaton construction

The plant automaton construction is based on rules and relations which is carried out according to 4 following steps : 
1. To build the automaton with \( 2^n \) states (\( n \) is the number of controllable events) describing all the possible evolutions of the controllable events starting from the initial conditions.
2. To add to this automaton the uncontrollable events resulting from occurrence rules,
3. To build a precedence automaton starting from the precedence relations and initial situation between the uncontrollable events,
4. To make the synchronous cross product of the automaton resulting from the occurrence rules and the precedence automaton.

This method of modelling is carried out for each part of the system. The complete model of plant is obtained then by asynchronous composition of all the elementary models.

For the horizontal movement, the one step consists in establishing a 4 states automaton. The evolutions between states are related to activation or deactivation of the \( GO\_RIGHT \) and \( GO\_LEFT \) orders (fig. 2.a). The second step respects the occurrence rules defined by the user and permits to supplement this automaton by the uncontrollable events \( left \) and \( right \) (fig. 2.b). With the precedence relations, the third step gives the second automaton

![Fig. 1. Characteristics of the pneumatic gripper orders must be observed. The \( GO\_RIGHT \) order activation makes possible the displacement of the cylinder to move towards the line and to leave the left position. The chronology of the uncontrollable events is the following one : deactivation of \( left \) then activation of \( right \). The parameters are presented in table 1.](image-url)
3. PRINCIPE OF OFF-LINE SYNTHESIS

The synthesis framework is based on the use of a controller and a supervisor. A dedicated intersection algorithm extracts the sequences of events that can be generated both by the controller, GSS, and by the supervisor, SUP. This algorithm results in an event-based automaton SYNC. A reduction algorithm allow to remove the deadlocks of SYNC and to generate the control-execution model.

The intersection step generate the automaton SYNC representing the behaviour common to the automata SUP and GSS. The evolutions of the control model, GSS, are authorised by the supervisor, SUP. Each state in GSS is associated with the orders of the corresponding situation of Grafcet. The transitions of GSS are given in terms of logical expressions combining input variables and their edges. On the other hand, SUP is an elementary automaton whose transitions correspond to simple events. The intersection algorithm goes through the events that can be accepted by GSS and SUP to construct SYNC. Only the control evolutions that are acceptable by the supervisor are retained in automaton SYNC. The states of SYNC relating transitions are grouped into a region, which corresponds to the given Grafcet situation. The intra-region transitions correspond to controllable events and the inter-region transitions represent uncontrollable events. The interleaving of parallel orders, inside a region, is used here to provide an event-based semantics to SYNC. This, in view of simplifying the formal development of the intersection procedure.

The reduction step is used to generate the maximum reactive non-blocking execution controller by removing all the blocking situations from SYNC. The resulting automaton (OPT) represents the most permissive sub-set of the behaviour of Grafcet. OPT is both non-blocking, and satisfies the given safety and liveness specifications. Each state of this automaton corresponds to an aggregation of the states of a region of SYNC after the removal of blocking evolutions. The states of OPT correspond to the stable situations of the controller to be implemented. They contain information about the active Grafcet steps. The outputs are simultaneously activated and deactivated in the corresponding situation. The transitions of OPT correspond to uncontrollable events whose effect is to drive the controller from one stable situation to another. In the worst case, the size of the automaton OPT is proportional to $|\text{GSS}| \times |\text{SUP}| \times 2^{|\text{OPT}|}$, where $|\text{GSS}|$ and $|\text{SUP}|$ represent the number of states of the automaton GSS and SUP respectively.

Here, the off-line synthesis approach results in an exhaustive control implementation model. This one requires a very large memory in the case of complex systems. An alternative on-line synthesis approach has therefore been proposed to reduce the size of such an exhaustive model. The on-line

Fig. 2. Horizontal movement automaton

Presented figure 2.c. Each state of this automaton authorizes all the events related to the plant element model. A synchronous cross product between the automaton resulting from step 2 and the precedence automaton resulting from step 3 allow to obtain the final automaton (fig. 2.d). The movement is then described in a complete way independently of the controller through 12 states and 35 transitions.

The automata of the vertical movement and aspiration movement are established in an identical way. The plant automaton is obtained by asynchronous product cross of these elementary models and corresponds to an automaton of 288 states and 2184 transitions.

2.3. Modelling of the constraints

We can use this methodology for the modelling of the constraints. We need only to extend the definitions of the occurrence rules and the precedence relations given previously. The occurrence rules define a link between the controllable events and the uncontrollable events and the occurrence rules can be “cause” or “consequence”. The precedence relations relate the same types of events. (Philipott et al, 2004).

We defined two types of constraints. The safety constraints between controllable events themselves use directly precedence relations. The liveness constraints establish a link between controllable events and uncontrollable events. Consequently, the modelling depends on the type of constraints to model.

The plant and constraints models are used to obtain the automaton SUP. This automaton corresponding to the supremal language of the supervised plant by applying the synthesis algorithm proposed in (Kumar, 1991). The transition structure of SUP represents the maximum permissible behaviour of the controlled plant with respect to the given constraints. At the same time, the automaton of stable situations, GSS, of Grafcet, deterministic and reactive, is extracting by applying the algorithm given in (Roussel, 1994).
synthesis proceeds by successive calculations of a partial control implementation model for a temporal window.

4. ON-LINE SYNTHESIS

In 1992, (Chung, et al., 1992) implements an on-line scheme which is basis of an N-step ahead projection of the behaviour of the process. For us, the on-line synthesis is based on the three steps of the off-line approach among which the first (modelling, synthesis and extraction) is identical to the off-line. Consequently, the procedure of control generation is based on a partial intersection step between \( \text{SUP} \) and \( \text{GSS} \) for a number of anticipated future evolutions. The step of treatment of blocking situations results from this partial intersection. A partial automaton represents the anticipated control implementation model of anticipated future evolutions.

4.1. Principle of the partial intersection.

The partial intersection procedure consists to construct the common behaviour of \( \text{GSS} \) and \( \text{SUP} \) starting from their current state from anticipated future evolutions. The method give in result an automaton \( \text{SYNC}^N \) where \( N \) represents the step of calculation. \( N \) is the future possible reactions of the plant from current situation of the system. For a partial intersection window where the size is limited by the value of \( N \), the automaton \( \text{SYNC}^N \), like \( \text{SYNC} \), is constituted by regions, intra-regions transitions and inter-regions transitions. The \( \text{SYNC}^N \) construction is in real time on a limited part of the controller and gives an unfinished automaton (fig. 3). Thus, while being interested only in the number of uncontrollable events \( \Sigma_w \), the automaton \( \text{SYNC}^N \) is constructed compared to futures reachable situations of Grafcet.

Consequently, for a region reached, all actions associate to the corresponding Grafcet situation are taken into account. The states of \( \text{SYNC}^N \) with no output transition are blocking states.

4.2. Formalization of the partial intersection

The automaton \( \text{SYNC}^N \) is defined by a 9-uplet \((\Sigma, \text{ST}, \text{TR}, \text{ST}, \text{TR}, f_0, \text{BLOC}, \text{EM}, \text{RD})\) with:

- \( \Sigma = \Sigma_t \cup \Sigma_u \), \( \Sigma_t = \uparrow \Sigma \cup \downarrow \Sigma \) et \( \Sigma_u = \uparrow \Sigma \cup \downarrow \Sigma \). In the continuation, \( \uparrow \Sigma \) corresponds to an element of \( \uparrow \Sigma \), \( \downarrow \Sigma \) with an element of \( \uparrow \Sigma \), \( \uparrow \Sigma \) with an element of \( \uparrow \Sigma \) and \( \uparrow \Sigma \) with an element of \( \uparrow \Sigma \).

- \( \text{ST} \) is the whole of the states of \( \text{SYNC}^N \). \( \text{ST} \) is partitioned in subsets called regions. A state \( \text{st} \in \text{ST} \) corresponds to a single configuration of the 3-uplet \((q, y, E)\). \( q \) is a state of \( \text{SUP} \), \( y \) a state of \( \text{GSS} \) and \( E \) the whole of the logical values (0 or 1) of the inputs of Grafcet. All the states corresponding to a situation of Grafcet and a single configuration of its inputs variables are grouped in a region. These regions can be grouped in a window of partial intersection whose size depends on \( N \).

- \( \text{TR} : \text{ST} \times \Sigma \rightarrow \text{ST} \) is a partial function representing the transitions from \( \text{SYNC} \). A transition is defined by a triplet \((\text{st}, \sigma, \text{st'})\) in \( \text{TR} \).

- \( \text{st}_0 \in \text{ST} \) is the initial state which is given by \((q_0, y_0, E_0)\). \( q_0 \) is the initial state of \( \text{SUP} \), \( y_0 \) the initial state of \( \text{GSS} \) and \( E_0 \) corresponds to the whole of the initial values of the inputs of Grafcet.

- \( r_0 \subset \text{ST} \) is the initial region corresponding to the initial situation \( y_0 \) of \( \text{GSS} \) and to the initial values \( E_0 \) of its inputs variables. This region includes the initial state \( \text{st}_0 \).

- \( \text{BLOC} \subset \text{ST} \) contains the whole of the blocking states, i.e. those not having outgoing transitions.

- \( \text{EM} \subset \text{ST} \) contains the whole of the states and the marked regions.

- \( \text{RD} \subset \text{ST} \) contains the whole of the already developed regions.

The automaton \( \text{SYNC}^N \) is generated by means of the following iterative algorithm (Philippot, 2002). When no window of intersection can be created anymore, the procedure is terminated.

Step 1: To develop the window of initial partial intersection \( f_0 \) starting from the initial region \( r_0 \).

Step 2: To develop a window of partial intersection unspecified \( f_r \).

This algorithm receives as an input the automaton \( \text{SUP} \), \( \text{GSS} \) and the initial structure of the automaton \( \text{SYNC}^N \), given by \((\Sigma, \{\text{st}_0\}, \emptyset, \text{st}_0, \{\text{st}_0\}, \emptyset, \emptyset, \emptyset)\). \( \text{SUP} = (\Sigma, Q, \Delta, q_0) \) corresponds to the supervisor generated by the synthesis step. The automaton \( \text{GSS} = (E, S, Y, T, y_0) \) is the equivalent of the Grafcet. In the following, the current state of \( \text{SYNC}^N \) is characterised by the current state of \( \text{SUP} \), the current situation of \( \text{GSS} \), as well as the current valuation of the input variables.

The iterative procedure corresponding to step (1) is given in figure 4. This procedure creates the states and the transitions of the initial region \( r_0 \). In the second step (2), as long as the number of uncontrollable events running \( \text{Ncou} \) is lower or equal to \( N \), it is about:

A) To develop the inter-regions transitions and to create the first state of each regions downstream. If \( \text{Ncou} = N-1 \) then the states and the regions upstream of the transition are safeguarded in the set of marked states \( \text{EM} \). The regions downstream are gathered in the set of developed regions \( \text{RD} \) to avoid the developments for the next step.

B) To safeguard the states blocking in \( \text{BLOC} \).
C) To develop each region downstream according to a principle similar to ①, these new regions belong to the initial window of partial intersection. The algorithm associated at the development of an unspecified window breaks up into 2 steps (Fig. 5).

The first step (②) consists in recovering the information already developed in the preceding partial intersection. It is about:

A) To recover the states and the regions marked in the intersection preceding and pertaining to EM.

B) To recover the already developed regions assemble in RD.

C) To initialise sets EM and RD for the future safeguards for the development of another unspecified window f_t.

The second step (③) corresponds to ② algorithm of development of the initial window (Fig. 4).

Fig. 4. Development of the initial window f_0

4.3. Treatment of deadlocks and generation of the automaton of partial controller

By analogy with the off-line synthesis, the principle of treatment of deadlocks consists in withdrawing the blocking states of \( \text{SYNC}^N \). These states are identified by the set BLOC, as well as the transitions making it possible to reach these states. The removal of these blocking evolutions can possibly generate a controller for a number of evolutions lower than \( N \). This step makes it possible to generate the automaton of partial controller \( \text{OPT}^N \).

The reduction of the size of the automaton of controller \( \text{OPT}^N \) is significant compared to an off-line approach. In the case of an on-line synthesis, in worst case, the number of states proportional to \((\Sigma_0)^2N\). For complexes systems, the profit memory capacity is then considerable. Concretely, as soon as the process generates an output, a new partial intersection/reduction is developed for a step moreover, and the automaton is updated.

Consequently, the architecture of establishment rests on the execution in parallel of two modules. The first module of on-line calculation of the controller carries out during the execution of the controller, a development for a step moreover. The second module of execution of the elaborate controller. Thus, the computing time of \( \text{SYNC}^N \) and \( \text{OPT}^N \) influences the least possible the reactivity of the order. However, in spite of the optimisation of the size memory, the on-line synthesis does not guarantee the absence of deadlocks during operation. Indeed, since a partial intersection takes into account only one limited window of future evolutions of the system, no detectable deadlocks in the current window can occur at the time of the following partial intersections. By increasing size \( N \) of the window, these possible blocking states will be avoided or detected sufficiently early to be able to anticipate actions of correction. Consequently, the choice of \( N \) constitutes a compromise between
the risk of deadlocks and the memory capacity necessary to the establishment of the elaborate controller.

The objective is now to establish a comparison between off-line and on-line synthesis for highlighting the profit in memory capacity. The table 2 recapitulates the sizes of the various automata for the pneumatic gripper.

<table>
<thead>
<tr>
<th>GSS</th>
<th>Process</th>
<th>Constraints</th>
<th>SUP</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 states</td>
<td>288 states</td>
<td>64 states</td>
<td>96 states</td>
</tr>
</tbody>
</table>

Table 2. Result of the various automata

For the simple example of pneumatic gripper, the step of generation of the order calculated off-line provides an automaton SYNC made up of 106

<table>
<thead>
<tr>
<th>Off-line synthesis</th>
<th>On-line synthesis</th>
<th>1st partial intersection</th>
<th>2nd partial intersection</th>
<th>3rd partial intersection</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYNC</td>
<td>SYNC</td>
<td>6 states, 4 transitions 6 states, 6 transitions</td>
<td>31 states, 10 transitions</td>
<td></td>
</tr>
<tr>
<td>OPT</td>
<td>OPT</td>
<td>4 states, 3 transitions 5 states, 5 transitions</td>
<td>6 states, 6 transitions</td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Automata generated off-line and on-line transitions and 52 states grouped in 30 regions. The controller automaton OPT includes 30 states and 62 transitions. For a window of N=2 steps, the first partial intersections is given table 3.

We represented the optimal controller (OPT) in the figures 6.a and 6.b. In each state is associated the set with the orders to send (ORD) and to prohibit by OPT (PROH). These sets are used to inform the user about the orders, which are maintained, or not by the synthesis procedure. The first region of SYNC corresponds to state 0 of OPT (fig. 6.a) where no order is sent. The two states of the second region are aggregate in a single state 1 where the order sent during the execution is GO_DOWN, no order is prohibited.

Fig. 6. Partial automata of controller (OPT)

The increase in the size of the window generates an increase in the number of states and transitions. Also a reduction in the number of intersection is a risk of reappearance of the problems involved in the off-line synthesis. A study on the choice of value N remains to be developed.

5. CONCLUSION

This paper presents an approach for off-line and on-line synthesis of an optimal control implementation. This method must respect a number of safety and liveness constraints for a given Grafcet. The resulting controller can be viewed as a programing executing Grafcet in accordance with deterministic and reactive semantics.

Our current research proposes to enhance the practicability of the proposed formal framework. This, by adapting a dedicated assistance methodology for the modelling of the plant and an extension for the constraints. The user specifies the occurrence rules and the precedence relations. These rules are automatically translated into automata compatible with the supervisory theory.

For complex systems, the off-line synthesis approach requires a "huge" memory size for the implementation. We present the formalization of the on-line approach which overcomes this limitation. However, deadlocks are not guaranteed to be avoided in this case. Our future work must implement an heuristic to choose a pertinent value for the number of required evolutions. This, in the aim to avoid deadlocks and to guarantee a maximum "acceptable" size of the partial implementation.

6. REFERENCES


Acknowledgments: A first version of this paper was published at IFAC Workshop on Discrete Event Systems, WODES’04, 22-24 September 2004.