Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

A Modular Synthesis Approach for Distributed Safety Controllers, Part B: [⋆] Modular Control Synthesis

Dirk Missal and Hans-Michael Hanisch [*]

[*] *Martin Luther University Halle-Wittenberg,*
Institute for Computer Science
06099 Halle, Germany
(Dirk.Missal, Hans-Michael.Hanisch)@informatik.uni-halle.de

Abstract: The contribution provides an approach for formal synthesis of controllers that ensure safe operation on the shop floor level. It is structured into two parts.
Part B presents the modular synthesis approach. It is based on the modular backward search in order to avoid the complexity of generating all states and state transitions of the plant model. It therefore uses modular backward steps that describe the trajectories leading to forbidden states. The generation of these trajectories is stopped as soon as a controllable (in our case preventable) step is found. From this information the models of the controllers are generated. Each controller has decision functions and communications functions. Together they establish a network of local, interacting controllers with communication. Up to now, we suppose that the plant is completely observable, i.e. the local controllers have complete information of the local states of the partial plants they are supposed to control. The method is illustrated by taking the example from Part A.

1. INTRODUCTION

The complexity is a major obstacle for application of synthesis to real scale problems. Therefore algorithms omitting complete enumeration of the state space have to be developed. The presented backward search works over symbolic markings given from a state predicate instead of a complete set of reachable markings. Hence, the implicit state representation of the model is used in the algorithm. Furthermore, modular approaches provide the opportunity to reduce complexity by distributing the global problem into a set of local subproblems. Modularity, in our case, is obtained by applying the synthesis algorithm on plant model parts defined by modules. The reduction potential is practically shown in the comparison of the example with the monolithic synthesis approach in Missal and Hanisch (2006) in Sec. 4.

Based on the fundamentals described in Part A Missal and Hanisch (2008), the modular synthesis of distributed safety controllers is described in this contribution. The starting point for synthesis is a well-structured modular *safe Net-Condition/Event system* ($_SNCES$) plant model and a formal specification in terms of forbidden state predicates. The model has to be partially composed. Forbidden state predicates are distributed before they can be used for a modular approach.

The goal is to synthesize a collaborating set of local controllers. Thereby every controller has local observation and controllability to disjoint plant parts. To ensure global specifications, the controllers exchange information via Boolean communication variables (see Fig. 1).

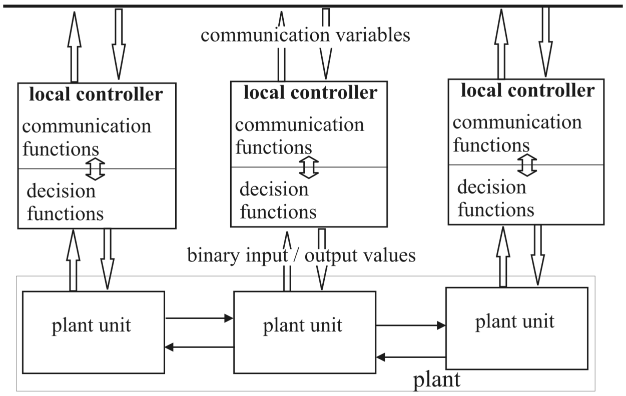

Fig. 1. Structure of the controlled system

In principle, we use the symbolic backward search as described for the monolithic synthesis approach in Missal and Hanisch (2006). But the possible steps for the modular approach are defined within basic modules and under consideration of the input state. The backward search operates just within one module while the dependencies with the modules are handled parallel. The modular backward search is described in Sec. 2. Section 3 addresses the whole synthesis process including the dependency handling. Finally a synthesis example is given in Sec. 4. A summary and a critical view of current shortcomings and limitations followed by providing directions for further research conclude the contribution.

[⋆] The work is supported by the Deutsche Forschungsgemeinschaft (DFG) under reference HA 1886/16-1 and HA 1886/16-2.

978-1-1234-7890-2/08/$20.00 © 2008 IFAC 10.3182/20080706-5-KR-1001.1006

Notations and definitions of Part A are used in the following sections without further reference.

2. MODULAR BACKWARD SEARCH

The backward search algorithm determines the complete set of uncontrollable trajectories leading to a forbidden state. A detailed description of the general idea of backward search is given in Missal and Hanisch (2006). For the modular synthesis the algorithm has to generate local prepredicates to forbidden state predicates. Then we analyse local possible steps and generate the prepredicates enabling the possible steps. If the analysed local step is uncontrollable, the prepredicates are identified as forbidden states too.

First the possible steps and local possible steps are defined based on the modular step definition given in Part A with Def. 3.5 and 3.7. Possible steps are steps defined under consideration of partial markings in terms of state predicates. They are called *possible* steps because the enabling conditions of enabled steps (see Part A Def. 3.4 and 3.6) can not be determined without consideration of concrete markings.

The local possible steps $\xi^p_{M_i}$ are defined at a module M_i. Further the possible steps $\xi^p_{ZP_i}$ are defined as the aggregation of local possible steps and depending on a local state predicate ZP_{M_i}. We define them as follows:

Let N be a $_SNCES$, Ξ_M a set of local steps and ZP_M a local state predicate. A local step ξ_M (following Def. 3.5 at Part A and not an enabled local step) is a *local possible step* $\xi^p_{M_i}$ iff the following holds:

$\nexists t' \in \xi^p_{M_i}$ for which holds:

- $\forall p \in P : (p, t') \in F | (m(p') = 1) \in ZP_M$ and
- $\forall p' \in P : (p', t') \in CN | (m(p') = 1) \in ZP_M$

Transitions satisfying a condition of the definition of $\xi^p_{M_i}$ cannot be part of a step from a marking $m(M)$ with $ZP(m(M)) = 0$ to a $m'(M)$ with $ZP(m'(M)) = 1$. A transition satisfying the first condition would remove a marking from a place $(m(p) = 1) \in ZP$ or restore the marking of a place $(m(p) = 0) \in ZP$ and the follower marking would not satisfy ZP. A transition satisfying the second condition is disabled because the source place of the condition arc is not marked.

Based on these local possible steps we define possible steps on depending modules of the whole modular plant model. For possible steps we have to ensure mostly the same conditions as for steps. But in difference to Def. 3.7 at Part A, we cannot define enabled possible steps because we do not consider any marking on the whole plant. Possible steps are defined as follows:

Let N be a $_SNCES$ and ZP_M a local state predicate, a step ξ (following Def. 3.6 at Part A and not an enabled step), consisting of local possible steps, is a *possible step* ξ^p iff the following holds:

Every

$$\xi^p_{ZP} = \bigcup_{M_n \in N} \xi^p_M$$

is a possible step if:

- for the local possible steps ξ^p_M holds $\exists \{ek\} \subseteq EK : \forall (t, e^{out}) \in EO^{arc} \wedge e^{out} \in \{ek\} | t \in \xi^p_{ZP}$ which *event input enables* [1] all $t^t_M \in \xi^p_M$ and
- $\exists \xi^p_M \in \xi^p_{ZP} : |\xi^p_M \cap T_{ZP}| \geq 1$, while $T_{ZP} = \{t^{pre}_{p_n} \in T : (m(p_n) = 1) \in ZP_M \wedge (t^{pre}_{p_n}, p_n) \in F\} \cup \{t^{post}_{\overline{p_n}} \in T : m(p_n) = 0) \in ZP_M \wedge (p_n, t^{post}_{p_n}) \in F\}$

It can be proven that the set of possible steps to a state predicate ZP includes all transition sets which can be part of an enabled step from a marking $m(N) : ZP_M(m(N)) = false$ to a marking $m'(N) : ZP_M(m'(N)) = true$.

We define possible steps depending on a predicate ZP_M because the following described backward search algorithm is defined on state predicates instead of a marking $m(N)$ of the net N.

The prepredicates ZP'_M for every possible step belonging to a predicate ZP_M are calculated due to the following rules:

The *re-determination rule* for ZP'_M is:

- $\forall p \in P : \exists t \in \xi^p_M : (p, t) \in F \Rightarrow ZA' = (m'(p) = 1) \in ZP'_M$
- $\forall p \in P : \exists t \in \xi^p_M : (t, p) \in F \Rightarrow ZA' = (m'(p) = 0) \in ZP'_M$, notation $\overline{ZA'}$
- $\forall p \in P : \exists t \in \xi^p_M : (p, t) \in CN \Rightarrow ZA' = (m'(p) = 1) \in ZP'_M$.

The statements follow from the marking and condition enabling condition for transitions (Def. 3.4 at Part A) and the definition of enabled steps.

In difference to the monolithic approach in Missal and Hanisch (2006) we need additionally a rule for consideration of condition couplings.

Therefore we define the *condition enabling rule* for all ZP'_M:

$\forall p \in P$ with $\exists ck \in CK : (p, c^{out}) \in CO^{arc} \wedge c^{out} \in ck | \exists t \in \xi^p_{M'} : (c^{in}, t) \in CI^{arc} \wedge c^{in} \in ck$

$\Rightarrow ZP'_M \wedge ZA(m(p) = 1) = ZP'_M$ or
if $\nexists ZP'_M \Rightarrow ZP'_M = ZA(m(p) = 1)$.

We have to ensure that transitions are condition enabled by condition inputs too, and therefore by the source places of condition interconnections. Condition inputs without connection to a source place are not considered by the rule. They are necessary for definition of controllable transitions.

Further the following *propagation rule* has to hold:
For ZA or $\overline{ZA}$ a state atom of ZP_M and ξ^p_M a local possible step from ZP'_M to ZP_M the atom ZA / $\overline{ZA}$ is in ZP'_M too, iff for p holds:

$$\nexists t \in T\, with\, (p, t) \in F : t \in \xi^p_M \wedge (m(p) = 0) \in ZP_M \text{ and}$$

$$\nexists t \in T\, with\, (t, p) \in F : t \in \xi^p_M \wedge (m(p) = 1) \in ZP_M$$

The predicate ZP_M can be empty for local possible steps. This holds for example if the local possible step is within a possible step and enables a local possible step at another module.

[1] see Part A, Def. 3.6

The predicate ZP'_M has to be free of contradictions. Thus ZP'_M has to have the following *consistency property*:

It must not exist two state atoms ZA_1 and ZA_2 within ZP_M which contradict each other, i.e. it must hold:

$$ZA_1 = (m(p) = a) \wedge ZA_2 = (m(p) \neq a)$$ [2]

Possible steps enabled by contradictive prepredicates are not taken into account further on. Reachable states satisfying contradictive predicates cannot exist and therefore possible steps to contradictive prepredicates cannot be part of an enabled step.

To improve the efficiency of the synthesis algorithm we calculate place invariants of all Petri net parts (see Missal and Hanisch (2006)) within the net. Prepredicates containing invariants are not treated further on.

Under consideration of the mentioned rules we determine the local prepredicates ZP'_M. Thereby prepredicates are generated for all local possible steps included in the possible step belonging to ZP.

Definition 2.1. Let N be a ${}_SNCES$, ZP_M a local state predicate and ξ^p_{ZP} a possible step over N. For every local possible step $\xi^p_M \subseteq \xi^p_{ZP}$ the set of prepredicates $PZP_M(ZP_M, \xi^p_{M,i})$ of ZP_M is the set of state predicates ZP'_M, which is calculated by backward analysis of steps ξ^p_M under consideration of the re-determination rule, the condition enabling rule, the propagation rule and the consistency property.

□

The generated predicates together with the local possible steps can be represented as graphs called *local backward graph*, where $BG_{n,i} = (CP_{M_i}, BA)$ is the nth graph within the module M_i. The nodes are $CP_{M_i} = (k, ZP, CO)$ with the index k, the local state predicate ZP_{M_i} and an associated communication predicate CO. The arcs are $BA = \{ZP_{M_i}, \xi^p_{M_i}, ZP'_{M_i}\}$, while $\xi^p_{M_i} \subseteq \xi^p_{ZP}$. For every newly assigned prepredicate it is checked whether there already exists a node with the same predicates. If not, a new node and depending arc are created. Thereby CO is inherited from ZP_{M_i} the local possible step is built from, if it exists. A backward step to an already existing node is considered within CO of the existing node in a way described later on.

To consider the dependency between the local steps within the local predicates generated by the same possible step, we additionally generate communication variables com_l, where l is an index. We define two communication variables, com_l^+ and com_l^- for every event interconnection necessary for building the possible step. That means that one pair for every ek satisfying condition 2) of the definition of possible steps is generated. A communication variable representing communication in direction of the event interconnection ek is symbolised with com_l^+, and the one in reverse direction with com_l^-, as displayed in Fig. 2. A pair of communication variables belonging to an ek is symbolised by $com_l \rightleftharpoons ek$.

The communication variables are considered in the local backward graph, more precisely in the communication predicate CO. The predicate consists of all communication variables related to that control predicate. For every possible step we define *local communication predicate components* $coc^{M_i}_{\xi^p}$.

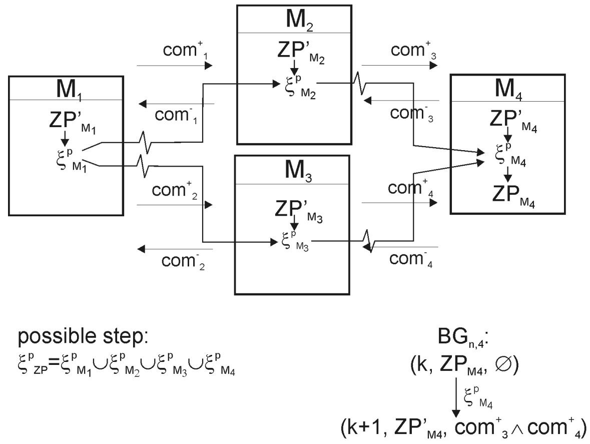

Fig. 2. Example scheme of a modular backward step and it results

A $coc^{M_i}_{\xi^p}$ is built $coc^{M_i}_{\xi^p} = coc^{M_i}_{\xi^p} \wedge com_l$

- for all com_l^+ for which holds: $\exists t \in \xi^p_{ZP} : (t, e^{out} \in EO^{arc}_{M_j} \wedge e^{out} \in ek \wedge \exists t^t_{M_i} \in \xi^p_{ZP} : (e^{in}, t^t_{M_i}) \in EI^{arc}_{M_i} \wedge e^{in} \in ek \wedge com_l^+ \rightleftharpoons ek$
- for all com_l^- for which holds: $\exists t \in \xi^p_{ZP} : (t, e^{out} \in EO^{arc}_{M_i} \wedge e^{out} \in ek \wedge \exists t^t_{M_j} \in \xi^p_{ZP} : (e^{in}, t^t_{M_j}) \in EI^{arc}_{M_j} \wedge e^{in} \in ek \wedge com_l^- \rightleftharpoons ek$

Every follower node inherits the CO of the source node belonging to the analysed possible step. The new local communication predicate to ξ^p_{ZP} is attached conjunctively $CO := (CO) \wedge coc^{M_i}_{\xi^p}$. The scheme of defining nodes and arcs of a local backward graph to a possible step ξ^p is shown in Fig.2.

If a local possible step $\xi^p_{m_i}$ leads to a local state predicate ZP for which already exists a node (k, ZP, CO), then the communication predicate of the source node, conjunctively connected with $coc^{M_i}_{\xi^p}$ resulting from $\xi^p : \xi^p_{M_i} \subseteq \xi^p$ is attached disjunctively to CO. The association of state predicates by communication variables and predicates is schematically illustrated in Fig.3. The step $\xi^p_{M,2}$ in Fig.3 is such a step backward leading to the existing node $(3, ZP_3, CO_3)$. Thus, the communication predicate $CO_3 = CO_1$ is extended by the communication predicate $coc\xi^p_{M,2}$ resulting of $\xi^p_{M,2}$ to $CO_2 \wedge coc\xi^p_{M,2}$. It has to be mentioned that only nodes with identical state predicates are melt that way. A node capturing a subset of states of another node has to be treated separately. Otherwise the control gets unnecessarily restrictive. In Fig. 3 the arcs are directed in the firing direction of steps. The backward search algorithms analyses the steps in reverse direction.

Generally, the composition of communication variables is deduced by a representation of backward search in a monolithic model. Every conjunction of communication variables composes together one predicate over the composed model. Disjunctive coupling represents different "global" predicates with the same local component. That relation

[2] while $a \in \{0; 1\}$

naturally follows as conversion from the proven rules for distribution of monolithic predicates presented in Missal and Hanisch (2006).

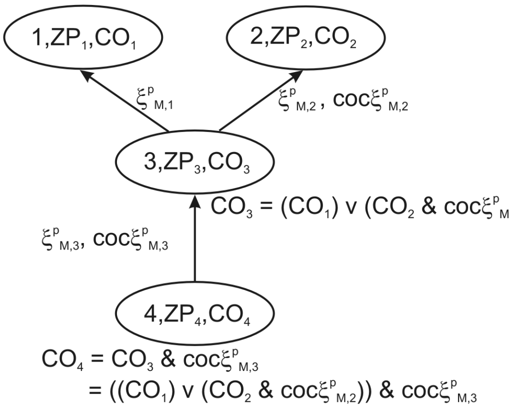

Fig. 3. Schematic example of the creation and the composition of communication predicates

If a communication predicate is extended because of a new connection within BG, we also have to extend the CO of all nodes reachable from the updated node (as can be seen at node four in Fig. 3). Reachable are all local state predicates (nodes) for which a trajectory of local steps exists $\xi^p_{M_i}$ in $BG_{M_i} : ZP \in BG_{M_i}$ leading to it.

The control functions of the distributed controllers are generated directly by analysing the local backward graphs.

The use of graphs including the communication predicates gives the possibility to use the results of already analysed local step trajectories for all forbidden states. Different forbidden states can depend on the same behaviour of the plant, therefore the analysis of different possible steps can be enabled by a local state predicate already analysed (ZP'). We check every generated local prepredicate for being already analysed to prevent repeated treatment. If such an already analysed predicate occurs, we generate an arc from the analysed local predicate (ZP) to the local prepredicate (ZP') at BG and modify the communication predicate of the affected nodes in the already described manner. That way two graphs can be melt to one if they contain local steps leading to the same prepredicate (see Fig.3).

The stop criterion for the backward search is the existence of controllable steps to a forbidden state predicate. Hence, for every calculated step it is checked if it needs condition inputs to be enabled for activating the step. Such steps can be prevented from firing by disabling the plant input, i.e. the step is controllable. If a calculated step is preventable by setting a condition input to zero, then the prepredicate is stored together with the locking inputs as decision predicates DP. The required condition input set ($CEM = \{ci^{in} \in C^{in} \,|\, \exists t \in \xi^p_M : (c^{in}, t) \in CI^{arc}\}$) for the local step from ZP'_M to ZP_M, resulting from the enabling rule of $_SNCES$, is identified therefore. If a marking satisfying ZP'_M occurs, then an input status is' would have to enable the step ξ^p_M for reachability of a marking satisfying ZP_M and for is' has to hold:
$\forall c^{in} \in C^{in} : \exists t \in \xi^p_M : (c^{in}, t) \in CI^{arc} \wedge \nexists c^{in} \in CK \Rightarrow is' = 1$.

The controller has to prevent the firing of ξ^p_M to avoid the forbidden state. The control function has to ensure:

> If the system is at a state satisfying ZP'_M, then $c^{in}_i \in C^{in} \,|\, \exists t \in \xi^p_M : (c^{in}_i, t) \in CI^{arc} \wedge \nexists c^{in} \in CK\}$, $is'(c^{in}_i) = 0$ has to be true for at least one input.

As control components result from preventable steps, the node to the prepredicate ZP'_M and an input to be influenced are stored at a set of decision components $DP = \{(CP_M, CEM)\}$. The whole synthesis process including the final extraction of the control functions is presented in the next section.

3. DISTRIBUTED CONTROL SYNTHESIS

In contrast to the monolithic synthesis (Missal and Hanisch (2006)) we perform the backward search module by module under consideration of the signal interconnection between them. The result is a distributed control structure with one controller for every unit module. Therefore the control structure is predefined by the modular structure of the plant model. The synthesis procedure runs through the following steps:

(1) composition of the hierarchical modular plant model to a modular structure of basic modules for the units,
(2) derivation of local specifications from a global specification,
(3) modular backward search
(4) derivation of the control functions from the backward search results.

These steps are described in more detail in the following.

A $_SNCES$ plant model can consist of multiple hierarchy levels. It is partially composed to a modular model of basic modules as described in Part A of this contribution. Every plant module represents the observed and controlled plant part for one controller.

Next the forbidden state predicates have to be *transformed to local state predicates*. The methodology is similar to the distribution of the control predicates presented in Missal and Hanisch (2006). If a specification predicate includes state information of more than one plant module, all state atoms related to the same module are cut out from the state predicate and form a local state predicate as defined in Def. 3.10 in Part A. If a local state predicate is assigned to more than one communication variable, the conjunction of the variables builds the CO. The assignment of communication variables can be done in any order under these conditions. The building of the CO is similar to the representation of event interconnection within a possible step described in Sec.2. An example for building local specifications is presented in the next section.

Using the local state predicates and the modular plant model the *backward search* is started as described in Sec.2. The backward search determines when for every prepredicate only controllable backward steps are calculated or no more prepredicates are determined.

When the backward search is completed we have to *generate the control functions* from the local backward graphs. Our distributed controllers consist of local decision functions defining the state of control outputs and communication functions defining the value of the communicated variables. We use the same structure as presented in Missal and Hanisch (2006) (Fig. 1). For the decision functions only nodes of BG's with a decision predicate DP are of interest. We define functions associated to every occurring plant input, one for every input. Because every plant part input is controlled by exactly one local controller we can build the function locally, i.e. by analysing the BG's of one module. Local decision functions $DF_{c^{in}}^{M_i}$ are defined for every local input c^{in}.

$$DF_{c^{in}}^{M_i} = \bigvee_{\forall ZP_{M_i} \in (CP_M, CEM): c^{in} \in CEM} (ZP_{M_i} \wedge CO_{ZP_{M_i}})$$

The decision function consists of local state predicates and communication variables. The communication variables represent a set of local state observations of other (i.e. not directly observable) plant parts. Ramadge and Wonham (1986) show that global predicates can be decomposed to conjunctively combined local predicates. Therefore, local state predicates are conjunctively combined with communication variables representing "external" state predicates.

Communication functions are generated for every communication variable included in a communication predicate. Before the functions are generated, the communication predicates CO are transformed to disjunctive normal form (DNF) CO^{DNF}. In the DNF of such communication predicate every conjunction represents a combination of communication variables defining a single system state while the disjunction represents different global states with the same local predicate. From that context results the following rule for generation of communication functions.

The communication function for com_l is a disjunctive composition of all state predicates associated with com_l within the CO. The state predicates are built from the local state predicates ZP_{M_i} conjunctively combined with a communication term. That term consists of the disjunctive combination of all conjunctive terms of CO including com_l, while com_l has to be removed of these terms.

$$com_l^s = \bigvee_{\forall BG_M : com_l^{\neg s} \in CO \wedge BG_M \notin DP} ZP_{M_i} \wedge \bigvee_{\forall coc : com_l^{\neg s} \in coc} coc \backslash com_l^{\neg s},$$

while coc are the conjunctive terms of CO^{DNF} and $s \in \{+, -\}$.

We get a set of local decision functions and communication functions for every plant part. These functions together form the distributed controllers. We get the same structure of local controllers and communication as presented in Missal and Hanisch (2006), but the functions are got from the backward search result directly. The number of communication variables needed is less or equal compared to the homogeneous approach. The modular approach can include chains of communication variables, i.e. communication functions can include communication variables. These advantages together with a reduced complexity of the backward search are pointed out using the following example.

4. SYNTHESIS EXAMPLE

The modular plant model and the formal specification introduced in Part A are used for the synthesis example in the following. The partially composed $_SNCES$ model of the ejection unit and the measuring unit is displayed in Fig. 4. The forbidden state predicate $p(ec\_nr) \wedge p(mc\_nr)$ is the used specification.

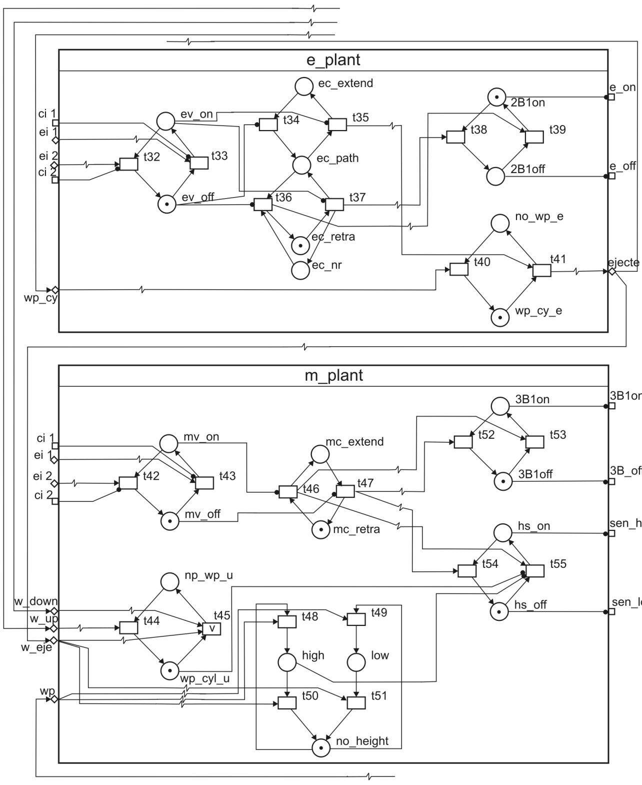

Fig. 4. Partially composed cutout of the sNCES model for the testing station

As described in Sec.3 the backward search is processed in a modular manner and we have to transform the forbidden state predicate first. The two state atoms are related to different modules and form local predicates each apart. Because they have to represent a global state together we link them by communication variables. The communication variables $com_1^+; com_1^-$ are included in the communication predicates CO of the state predicates. We get two root nodes for local backward graph in the modules:

Module	$CP_M = \{k, ZP, CO\}$
ejection	$\{1, \{p(ec\_nr)\}, (com_1^+)\}$
measuring	$\{1, \{p(mc\_nr)\}, (com_1^-)\}$

Starting from the local state predicates the backward search is performed. A marking of the place $p(ec\_nr)$ can be reached by the local possible step $\xi_{M_e}^p = \{t37\}$. The step is enabled by the state predicate $VZP = p(ev\_on) \wedge p(ec\_retra)$. It is a local step and has no dependence to any other local possible step. Therefore the communication predicate stays unchanged. The resulting node is $\{2, \{p(ev\_on) \wedge p(ec\_retra)\}, (com_1^+)\}$. Since the transitions of the step are not controllable we have to go on with the backward search. The next possible step is $\xi^p_{M_e} = \{t32\}$ in the ejection module. The enabling prepredicate is $VZP = p(av\_off) \wedge p(ec\_retra)$ and we save the node $\{3, \{p(ev\_off) \wedge p(ec\_retra)\}, (com_1^+)\}$. The transition $t32$ is controllable and the analysis of that local graph is finished because there are no other possible steps. The evaluation of the second local graph runs similarily. The results of the backward search are displayed in Fig.5.

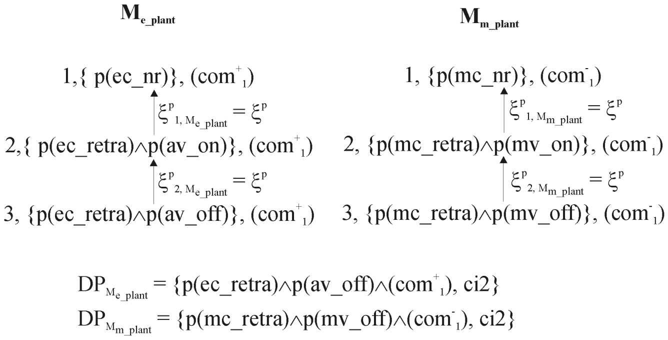
M_{e_plant}
M_{m_plant}
1,{ p(ec_nr)}, (com^+_1)
$\xi^p_{1, M_{e\_plant}} = \xi^p$
2,{ p(ec_retra)∧p(av_on)}, (com^+_1)
$\xi^p_{2, M_{e\_plant}} = \xi^p$
3, {p(ec_retra)∧p(av_off)}, (com^+_1)
1, {p(mc_nr)}, (com^-_1)
$\xi^p_{1, M_{m\_plant}} = \xi^p$
2, {p(mc_retra)∧p(mv_on)}, (com^-_1)
$\xi^p_{2, M_{m\_plant}} = \xi^p$
3, {p(mc_retra)∧p(mv_off)}, (com^-_1)
$DP_{M_{e\_plant}}$ = {p(ec_retra)∧p(av_off)∧(com^+_1), ci2}
$DP_{M_{m\_plant}}$ = {p(mc_retra)∧p(mv_off)∧(com^-_1), ci2}

Fig. 5. Backward search result for the forbidden state f3

Table 1. distributed control functions for the specification example

ejection controller		
cie 2 = 0	if	$ec\_retra \wedge ev\_off \wedge com_1^+$
$com_1^- = 1$	if	$ec\_nr \vee ec\_retra \wedge ev\_on$
measuring controller		
cim 2 = 0	if	$mc\_retra \wedge mv\_off \wedge com_1^-$
$com_1^+ = 1$	if	$mc\_nr \vee mc\_retra \wedge mv\_on$

These results are in the following compared to the homogeneous approach in Missal and Hanisch (2006). For this approach the same example is discussed. The homogeneous distributed controller synthesis approach works on a fully composed model. In comparison with the monolithic synthesis the number of analysed backward steps is reduced from 8 to 4. That reduction leads also to an reduced number of states (from 8 to 6). Four communication variables are generated with the monolithic approach as presented in Missal and Hanisch (2006), while the example above comes out with just two. A similar reduction can be seen on other examples not discussed in this contribution.

5. CONCLUSION

Control synthesis and especially distributed control synthesis are a major issue of scientific research and practical application. The reduction of computational complexity turns out to be the main requirement for real scale application.

We have shown that a modular approach based on the backward search offers major advantages. It reduces the complexity in terms of numbers of analysed steps and states and leads directly to compact distributed control functions. That is shown by a comparison with the homogeneous approach using backward search too. Both approaches avoid the enumeration of the complete reachability set.

Up to now, the method has neither been proven to be correct nor to be maximally permissive. A proof for maximal permissiveness of the monolithic procedure is there, but not yet published.

Hence, we see a large open field of questions that need to be answered in further work.

Despite of these previously unanswered questions, the proposed method, however, shows potentials that may bring synthesis a significant step towards feasibility to systems of realistic size and complexity.
The potential benefits are significant:

(1) The algorithm has less computational complexity than the one that works on complete composed plant models. The gain one could get obviously depends on the structure of the plant model and the assignment of controllers to it (e.g. grade of distribution). This, in turn, depends on the decision of human beings.
(2) The plant model is as close to reality as it can be. It can be constructed systematically from predefined modules rather than designed from the scratch.
This helps the human in the modelling and validation process.
(3) The presented synthesis approach can be run partially parallel. This aspect is remarkable and may become more interesting for further application to systems of realistic scale.

It is obvious that further work has to focus on the formal aspects mentioned above. For a more realistic methodology, also incomplete state observation has to be included in the research.

REFERENCES

D. Missal and H.-M. Hanisch. Synthesis of distributed controllers by means of a monolithic approach. In *Proceedings of the 11th IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA'2006)*, pages 356–363, September 2006.

D Missal and H.-M. Hanisch. A modular synthesis approach for distributed safety controllers, part a: Modelling and specification. In *Proceedings of the 17th IFAC World Congress*, Seoul, Korea, 2008.

P. J. Ramadge and W. M. Wonham. Modular supervisory control for discrete event systems. In *Seventh Internat. Conf. Analysis and Optimazition of Systems, Nice, France*, June 1986.