Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

MAS

Modelling, Analysis and Simulation

Modelling, Analysis and Simulation

Coalgebra and coinduction in discrete-event control with partial observations

J. Komenda, J.H. van Schuppen

REPORT MAS-E0422 DECEMBER 2004

CWI is the National Research Institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organization for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-3703

Coalgebra and coinduction in discrete-event control with partial observations

ABSTRACT
Coalgebra and coinduction provide new results and insights for the supervisory control of discrete-event systems (DES) with partial observations. The paper is based on the formalism developed for supervisory control of DES in the full observation case, i.e. the notion of bisimulation, its generalizations (partial bisimulation and control relation), and the finality of the automaton of partial languages. The concept of nondeterministic weak transitions introduced in this paper yields a definition of deterministic weak transitions. These are shown to be useful in the study of partially observed DES. They give rise to the relational characterizations of normality and observability. These characterizations lead to new algorithms for supremal normal and supremal normal and controllable sublanguages that are compared to the ones known in the literature. Coinduction is used to define an operation on languages called supervised product, which represents the language of the closed-loop system, where the first language acts as a supervisor and the second as an open-loop system. This technique can be used to define many important languages, e.g. supremal controllable sublanguages, infimal controllable or/and observable superlanguages. A variation of supervised product corresponding to the permissive control policy with full controllability is given. It is shown to be equal to the infimal observable superlanguage. We have obtained as a byproduct coinductive definitions of these important languages. We show that antipermissive control policy cannot be captured by coinduction. However, we present an algorithm based on the antipermissive control policy for the computation of an observable sublanguage that contains the supremal normal sublanguage. Using a similar method monolithic algorithms for computation of supremal normal and supremal normal and controllable sublanguages are developed. Finally, the lattice theoretic continuity of the supervised product (i.e. the distributivity of the supervised product with respect to partial language unions) is studied.

2000 Mathematics Subject Classification: 93C65, 93B25.
Keywords and Phrases: Discrete-event systems, partial observations, supervisory control, coalgebra, coinduction.
Note: This work was carried out under project MAS2-CONTROL. It was sponsored in part by the NWO project Coalgebra and control.

Coalgebra and Coinduction in Discrete-Event Control with Partial Observations

Jan Komenda
Institute of Mathematic, Czech Academy of Sciences, Brno Branch,
Zizkova 22, 616 62 Brno, Czech Republic
E-mail: komenda@ipm.cz

Jan H. van Schuppen
CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands,
Email: J.H.van.Schuppen@cwi.nl

Abstract

Coalgebra and coinduction provide new results and insights for the supervisory control of discrete-event systems (DES) with partial observations. The paper is based on the formalism developed for supervisory control of DES in the full observation case, i.e. the notion of bisimulation, its generalizations (partial bisimulation and control relation), and the finality of the automaton of partial languages. The concept of nondeterministic weak transitions introduced in this paper yields a definition of deterministic weak transitions. These are shown to be useful in the study of partially observed DES. They give rise to the relational characterizations of normality and observability. These characterizations lead to new algorithms for supremal normal and supremal normal and controllable sublanguages that are compared to the ones known in the literature.

Coinduction is used to define an operation on languages called supervised product, which represents the language of the closed-loop system, where the first language acts as a supervisor and the second as an open-loop system. This technique can be used to define many important languages, e.g. supremal controllable sublanguages, infimal controllable or/and observable superlanguages. A variation of supervised product corresponding to the permissive control policy with full controllability is given. It is shown to be equal to the infimal observable superlanguage. We have obtained as a byproduct coinductive definitions of these important languages. We show that antipermissive control policy cannot be captured by coinduction. However, we present an algorithm based on the antipermissive control policy for the computation of an observable sublanguage that contains the supremal normal sublanguage. Using a similar method monolithic algorithms for computation of supremal normal and supremal normal and controllable sublanguages are developed.

Finally, the lattice theoretic continuity of the supervised product (i.e. the distributivity of the supervised product with respect to partial language unions) is studied.

2000 Mathematics Subject Classification: 93C65, 93B25.
Keywords: Discrete-event system, partial observations, supervisory control, coalgebra, coinduction.

1 Introduction

Many different methods in discrete-event (dynamical) systems (DES) have been developed [5]. Only recently DES have been studied using coalgebraic techniques [20]. DES are mostly represented by automata viewed as a particular algebraic structure. They have been introduced by W.M. Wonham and his co-workers [23] and have been extensively studied since 1980's by many researchers, see e.g. [32], [14], [15], [3], [15], [33], [6], [18] etc. Supervisory control theory has been also extended to the study of infinite behavior of automata, see e.g. [28] and [29].

It is known that automata are at the same time algebras as well as coalgebras [19]. Therefore they can also be viewed as so called partial automata [20], which represent a particular instance of state-transition systems. Partial automata are coalgebras of a simple functor on the category of sets. Coalgebras are categorical duals of algebras (the corresponding functor operates from a given set rather than to a given set). In the last decade they have been extensively studied and used in the semantics of programming for infinite data structures (e.g. streams), while algebraic techniques have been used for dealing with finite data types as finite lists. Coalgebras have been found to be suitable in system theory as well for the description of the dynamic systems as deterministic automata and their various extensions (state transition systems, weighted automata, transducers etc.) The theory of universal coalgebras as a general theory of systems has been developed in analogy with the dual theory of universal algebra [21]. The algebraic notion of congruence relations has its coalgebraic counterpart: bisimulations relations. Final coalgebras are dual to initial algebras. The universal property of finality gives rise to the definition and proof principle called coinduction in the same way as induction is based on initiallity of an algebra. On a final coalgebra, bisimilarity coincides with equality, whereby proving equality of two elements of final coalgebras (e.g. languages) amounts to constructing a bisimulation relation that relates them.

A pioneering study of the relationship between controllability and bisimulation is presented in [2]. The authors have shown in Theorem 3.1 that controllability is equivalent to bisimilarity with respect to the set of uncontrollable events. This idea is a major motivation for application of coalgebra to discrete-event control and has been explored further in [20]. This paper presents a formulation of control of DES with partial observations in terms of coalgebra. Coalgebraic techniques are then applied to solve different problems in partially observed DES. The basic formalism is the one that has been developed by J.J.M.M. Rutten in [20], i.e. partial automata as models for DES and partial automaton of (partial) languages as the final coalgebra. The generalization to partially observed DES is not straightforward, but requires the development of new concepts. On the other hand, the basic ideas, i.e. relational characterizations of different notions and properties and the use of a powerful technique called coinduction are the strong points of the application of coalgebra to the control of DES. Indeed, the main advantage of the use of coalgebra is the naturally algorithmic character of the results, there is a canonical way how to check the properties like controllability or observability by constructing corresponding relations. Another advantage is the possibility of using the coinductive definitions and proofs that are shown to be useful in many situations. In particular, many extremal superlanguages or sublanguages can be defined by coinduction, which is an alternative to different type of formulas presented in e.g. [13], [24], and [3]. This yields also new algorithms for computation of these languages.

In the case of imperfect (partial) observations only a subset of events is actually observed by the controller. Being inspired by the theory of concurrency, we introduce in our setting the concept of weak transitions, including a deterministic concept, which is shown to be useful in the study of control problems with partial observations. It enables the definition of an auxiliary relation that corresponds to the observational equivalence and also observability relation which corresponds

to the observability of a language with respect to an open-loop system (language). A collection of necessary and sufficient conditions for a given partial language to be exactly achievable by a supervisory controller is formulated by relations called partial bisimulations and a coalgebraic formulation of the main theorem of supervisory control of DES with partial observations is presented.

Moreover, two different coalgebraic characterizations of normality are given, which can be compared to [7], where the algebraic characterizations using the concept of invariant relations have been presented. New algorithms for the computation of supremal normal (and normal and controllable) sublanguages that are based on these coalgebraic characterizations are proposed. Their computational complexity has been compared to that of known algorithms for computation of supremal normal sublanguages.

Coinduction is used to define a binary operation on languages called supervised product, which represents the language of the closed-loop system, where the first language is the language of a supervisor and the second language is that of an open-loop system. A minor modification of supervised product that disregards the controllability yields an operation that is shown to be equal to the infimal observable superlanguage. Using suitable automata representations a similar modification of the closed-loop language under the antipermissive control policy yields an observable sublanguage that contains the supremal normal sublanguage. A similar method yields new algorithms for computation of supremal normal and supremal normal and controllable sublanguages that are monolithic. A coinductive definition of the supremal controllable sublanguage is presented. The coinduction turns out to be well suited for formulating various concepts of discrete-event control.

The contribution of this paper when compared to the literature is twofold. Firstly, the whole framework of discrete-event control with partial observations introduced in [14] has been reformulated using the concepts from coalgebra and concurrency theory. This offers an additional insight to many problems and results. Secondly, our approach provides a refinement of the existing theory and yields new algorithms and useful concepts. Among new results novel algorithms for computation of supremal normal and supremal normal and controllable sublanguages have been proposed. Moreover some of them (Algorithm 5,6) are monolithic, which can be by itself considered as an important result. Unlike the known algorithms for supremal normal and controllable sublanguages, e.g. those developed by Cho and Marcus in [6] and by Yoo and Lafortune in [33]) that are iterations of two separate algorithms, our Algorithm 6 is compact. Such an algorithm is known to exist [11], but has not been explicitly presented. The concept of supervised product is central in our framework, because its coinductive definition describes in fact an event by event action of the supervisor. This considerably simplifies the study of properties of closed-loop languages compared to the classical algebraic approach.

The paper is organized as follows. Section 2 recalls the partial automata from [20] as the coalgebraic framework for DES represented by automata. The reader interested in more details about the key notions like bisimulation, coinduction, and finality should consult [21] or [20]. In Section 3 weak transition structures are defined on partial automata, powerset, projected, and observer automata are introduced using a deterministic notion of weak transitions. Observability relations are introduced in section 4 and normality relations in section 5. These relational characterizations are then used in section 6 to derive new algorithms for computing the supremal normal (and normal and controllable) sublanguages. Section 7 shows the power of the coinductive definition principle. After having specialized the observability relations to the final automaton of partial languages, necessary and sufficient conditions for a given language to be exactly achieved are captured in a relation called partial bisimulation. Finality is used to define the language of the closed-loop system as well as the infimal closed observable superlanguage. An algorithm is presented for the computation of an observable sublanguage that contains the supremal normal sublanguage. Mono-

lithic algorithms for computation of supremal normal (and normal and controllable) sublanguages are presented. Some preliminary results of this paper have been presented in [8] and [9] without proofs. The results of section 7 complete the algebraic results of [13], [24], and [7]. At the end of section 7 the distributivity of the supervised product with respect to basic language operations is studied.

2 Partial automata as coalgebras

Generators of discrete-event systems are just deterministic automata with logical outputs (that define the set of marked states) and partial transition function. They are defined in this section as coalgebras and corresponding notions of homomorphism and bisimulation are presented. In this section we recall from [20] partial automata as coalgebras with a special emphasis on the final coalgebra of partial automata, i.e. partial automaton of partial languages.

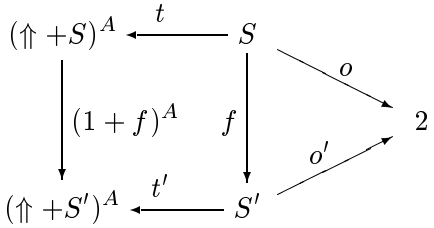
Before we recall the definition of partial automata we define $F-$coalgebras, where F is an endofunctor of the category Set with sets as objects and functions as morphisms. An $F-$coalgebra is a tuple $\langle S, \alpha\rangle$, where S is a set (also called carrier set) and $\alpha : S \to F(S)$ defines the coalgebraic structure on S.

Let A be an arbitrary set (usually finite and referred to as the set of inputs or events). The free monoid of words (strings) over A is denoted by A^*. The empty string will be denoted by ε. Denote by $\Uparrow = \{\emptyset\}$ the one element set and by $2 = \{0, 1\}$ the set of Booleans. A partial automaton is a pair $S = (S, \langle o, t\rangle)$, where S is a set of states, and a pair of functions $\langle o, t\rangle : S \to 2 \times (\Uparrow + S)^A$, consists of an output function $o : S \to 2$ and a transition function $S \to (\Uparrow + S)^A$. The output function o indicates whether a state $s \in S$ is accepting (or terminating): $o(s) = 1$, denoted also by $s\downarrow$, or not: $o(s) = 0$, denoted by $s\uparrow$. The transition function t associates to each state s in S a function $t(s) : A \to (\Uparrow + S)$. The set $\Uparrow + S$ is the disjoint union of S and $\Uparrow$. The meaning of the state transition function is that $t(s)(a) = \emptyset$ iff $t(s)(a)$ is undefined, which means that there is no $a-$transition from the state $s \in S$. $t(s)(a) \in S$ means that the $a-$transition from s is possible and we define in this case $t(s)(a) = s_a$, which is denoted mostly by $s \stackrel{a}{\to} s_a$. This notation can be extended by induction to arbitrary strings in A^*. Assuming that $s \stackrel{w}{\to} s_w$ has been defined, define $s \stackrel{wa}{\to}$ iff $t(s_w)(a) \in S$, in which case $s_{wa} = t(s_w)(a)$, also denoted by $s \stackrel{wa}{\to} s_{wa}$. It is easy to see that partial automata are coalgebras of the set functor $F = 2 \times (\Uparrow + (.)\ )^A$.

A *homomorphism* between partial automata $S = (S, \langle o, t\rangle)$ and $S' = (S', \langle o', t'\rangle)$ is a function $f : S \to S'$ with, for all $s \in S$ and $a \in A$:

$$o'(f(s)) = o(s) \text{ and } s \stackrel{a}{\to} s_a \text{ iff } f(s) \stackrel{a}{\to} f(s_a),$$

in which case: $f(s)_a = f(s_a)$.



A partial automaton $S' = (S', \langle o', t'\rangle)$ is a *subautomaton* of $S = (S, \langle o, t\rangle)$ if $S' \subseteq S$ and the inclusion function $i : S' \to S$ is a homomorphism. It is important to notice that the coalgebraic

concept of subautomaton corresponds to the notion of strict subautomaton in [6]. In the sequel we use always subautomata in the coalgebraic sense defined above, i.e. strict subautomata are meant.

Note that partial automaton as defined above is just a coalgebraic reformulation of what is understood to be a generator of a DES. Indeed, the transition function can be viewed in the coalgebraic form above, and the output function determines the subset of marked (or final) states (those whose output value is equal to 1).

A *simulation* between two partial automata $S = (S, \langle o, t\rangle)$ and $S' = (S', \langle o', t'\rangle)$ is a relation $R \subseteq S \times S'$ with, for all $s \in S$ and $s' \in S'$:

$$\text{if } \langle s, s'\rangle \in R \text{ then } \begin{cases} (i) & o(s) \leq o(s'), \text{ i.e. } s\downarrow \Rightarrow\ s'\downarrow, \text{ and} \\ (ii) & \forall a \in A:\ s \xrightarrow{a} \Rightarrow\ (s' \xrightarrow{a} \text{ and } \langle s_a, s'_a\rangle \in R), \end{cases}$$

A *bisimulation* between two partial automata $S = (S, \langle o, t\rangle)$ and $S' = (S', \langle o', t'\rangle)$ is a relation $R \subseteq S \times S'$ with, for all $s \in S$ and $s' \in S'$:

$$\text{if } \langle s, s'\rangle \in R \text{ then } \begin{cases} (i) & o(s) = o(s'), \text{ i.e. } s\downarrow \text{ iff } s'\downarrow \\ (ii) & \forall a \in A:\ s \xrightarrow{a} \Rightarrow\ (s' \xrightarrow{a} \text{ and } \langle s_a, s'_a\rangle \in R,) \text{ and} \\ (iii) & \forall a \in A:\ s' \xrightarrow{a} \Rightarrow (s \xrightarrow{a} \text{ and } \langle s_a, s'_a\rangle \in R). \end{cases}$$

We write $s \sim s'$ whenever there exists a bisimulation R with $\langle s, s'\rangle \in R$. This relation is the union of all bisimulations, i.e. the greatest bisimulation also called bisimilarity. It is immediate from the definition of bisimulation that two states are bisimilar iff they can make the same transitions and they give rise to the same outputs:

Proposition 2.1. *For any partial automaton $S = (S, \langle o, t\rangle)$ and any $s, s' \in S$:*

$$s \sim s' \textit{ iff } \forall w \in A^*:\ s \xrightarrow{w} \iff s' \xrightarrow{w}, \textit{ in which case } o(s_w) = o'(s'_w).$$

2.1 Final automaton of partial languages

In this subsection we define a partial automaton that is final among all partial automata and satisfies a proof principle called coinduction. The states of this final automaton represent minimal realizations of all possible behaviors (called partial languages) of all partial automata. Partial languages will be endowed with a (partial) automaton structure, which has the universal property of being final among all (partial) automata. The partial automaton of partial languages is defined using the Brzozowski notion of input derivative. Below we define the partial automaton of partial languages over an alphabet (input set) A, denoted by $\mathcal{L} = (\mathcal{L}, \langle o_{\mathcal{L}}, t_{\mathcal{L}}\rangle)$. More formally, $\mathcal{L} = \{\Phi : A^* \to (\Uparrow +2) \mid dom(\Phi) = \{w \in A^* \mid \Phi(w) \in 2\} \neq \emptyset \text{ is prefix-closed}\}$. To each partial language Φ a pair $\langle V, W\rangle$ can be assigned: $W = dom(\Phi)$ and $V = \{w \in dom(\Phi) \mid \Phi(w) = 1 (\in 2)\}$. Conversely, to a pair $\langle V, W\rangle \in \mathcal{L}$, a function Φ can be assigned : $\Phi(w) = 1$ if $w \in V$, $\Phi(w) = 0$ if $w \in W$ and $w \notin V$, and $\Phi(w)$ is undefined if $w \notin W$. Therefore we can write :

$$\mathcal{L} = \{(V, W) \mid V \subseteq W \subseteq A^*,\ W \neq \emptyset, \text{ and W is prefix-closed}\}.$$

The transition function $t_{\mathcal{L}} : \mathcal{L} \to (1 + \mathcal{L})^A$ is defined using input derivatives. Recall that for any partial language $L = (L^1, L^2) \in \mathcal{L}$, $L_a = (L^1_a, L^2_a)$, where $L^i_a = \{w \in A^* \mid aw \in L^i\},\ i = 1, 2$. If $a \notin L^2$ then L_a is undefined. Given any $L = (L^1, L^2) \in \mathcal{L}$, the partial automaton structure of $\mathcal{L}$ is given by:

$$o_{\mathcal{L}}(L) = \begin{cases} 1 & \text{if } \varepsilon \in L^1 \\ 0 & \text{if } \varepsilon \notin L^1 \end{cases}$$

and

$$t_{\mathcal{L}}(L)(a) = \begin{cases} L_a & \text{if } L_a \text{ is defined} \\ \emptyset & \text{otherwise} \end{cases}.$$

Notice that if L_a is defined, then $L_a^1 \subseteq L_a^2$, $L_a^2 \neq \emptyset$, and L_a^2 is prefix-closed. The following notational conventions will be used: $L\downarrow$ iff $\varepsilon \in L^1$, and $L \xrightarrow{w} L_w$ iff L_w is defined (iff $w \in L^2$).

Most of the rest of this section is recalled from [20].

Theorem 2.2. *$\mathcal{L}$ satisfies the principle of coinduction: for all K and L in $\mathcal{L}$, if $K \sim L$ then $K = L$.*

Proof. It follows from Proposition 2.1. Indeed, if $K \sim L$ then for any $w \in A^* : K \xrightarrow{w} \Leftrightarrow L \xrightarrow{w}$, i.e. $w \in K^2$ iff $w \in L^2$, in which case $o(K_w) = o'(K'_w)$, i.e. $w \in K^1$ iff $w \in L^1$. It follows that $K = L$. The converse implication is also true. □

Theorem 2.3. *The partial automaton $\mathcal{L} = (\mathcal{L}, \langle o_{\mathcal{L}}, t_{\mathcal{L}}\rangle)$ is final among all partial automata: for any partial automaton $S = (S, \langle o, t\rangle)$ there exists a unique homomorphism $l : S \to \mathcal{L}$. This homomorphism identifies bisimilar states: for $s, s' \in S$: $l(s) = l(s')$ iff $s \sim s'$.*

Proof. For the existence part of the theorem, we define the homomorphism l by putting for $s \in S$:

$$dom(l(s)) = \{w \in A^* : \ s \xrightarrow{w}\}$$

and

$$l(s) = ((l(s))^1, (l(s))^2) = (\{w \in A^* \mid s \xrightarrow{w} \text{ and } s_w\downarrow\}, \{w \in A^* \mid s \xrightarrow{w}\}).$$

Uniqueness of l follows from the fact that for any two homomorphisms $l, l' : S \to \mathcal{L}$ the relation

$$R = \{\langle l(s), l'(s)\rangle \in \mathcal{L} \times \mathcal{L} \mid s \in S\}$$

is a bisimulation. Therefore $l = l'$ follows from theorem 2.2. The last statement is immediate from the definition of l and Proposition 2.1. □

2.2 Coinduction

Coinduction is a dual concept to induction. Many people use induction without bearing in mind its abstract (categorical or universally algebraic) meaning. Coinduction in its full generality must be put into a general framework of universal coalgebra that uses the category theory. Finality of a coalgebra enables coinductive definitions and proofs in a similar way as initiality of an algebra enables definitions and proofs by induction. In order to make the paper more accessible to a reader not very familiar with category theory we have prefered to introduce the coinduction only in its special form: on final coalgebra of partial languages. It is the same as with mathematical induction that is by many people understood only on the initial algebra of natural numbers with the (unary algebraic) structure given by the successor operation: $\forall n \in N : \ succ(n) = n + 1$. Here definitions of functions by induction correspond to giving the successor on functions, hence yielding recursive formulas. Proofs by induction correspond to the very well known two-steps procedure, which amounts to verify that a relation is a congruence relation with respect to the successor operation. Similarly, a definition by coinduction amounts to give the corresponding structure, here output and derivatives on operations to be defined, and a proof by coinduction consists in verifying the conditions of bisimulation relation. We believe that giving a general categorical definition of coinduction would go far beyond the scope of the paper, which is only an

application of coalgebra to control. Coinduction has been well covered by the existing literature on universal coalgebra [21], [22].

Coinduction is used as a proof and definition principle throughout this paper. The use of coinduction is limited to final coalgebras. Behavior equivalence of two elements of final coalgebra means that these are equal. Also notice that the elements of final coalgebras are equal to their behaviors (the identity is the unique behavior homomorphism). This feature is sometimes paraphrased as 'being is doing', because these elements behave as they are.

Proofs by coinduction consist in constructing appropriate relations: for instance a proof of equality of two elements of a final coalgebra consists in finding a bisimulation relation that relates them. Definition by coinduction of an operation on elements of a final coalgebra consists in defining the same coalgebraic structure on the operation (for instance we define binary operations on partial languages by defining derivatives and output functions further in this paper). More details about coinduction and finality can be found in [21] or [20]. Various supervisory control and observation problems will be tackled using coinduction. It offers more then just an insight to some well known solutions of these problems, it leads to some new algorithms and results.

We adopt the notation from [19], page 9, easily extended from automata to partial automata, and denote the minimal (in size of the state set) representation of a partial language L by $\langle L\rangle$. Hence, $\langle L\rangle = (DL, \langle o_{\langle L\rangle}, t_{\langle L\rangle}\rangle)$ is a subautomaton of $\mathcal{L}$ generated by L. This means that $o_{\langle L\rangle}$ and $t_{\langle L\rangle}$ are uniquely determined by the corresponding structure of $\mathcal{L}$. The carrier set of this minimal representation of L is denoted by DL, where $DL = \{L_u \mid u \in L^2\}$. Let us call this set the set of derivatives of L. Inclusion of partial languages that corresponds to a simulation relation is meant componentwise. The prefix closure of an (ordinary) language L is denoted by $\bar{L}$. Some further notation from [20] is used, e.g. 'zero' (partial) language is denoted by 0, i.e. $0 = (\emptyset, \{\varepsilon\})$.

There is yet another important concept that will be needed in this paper. Namely, given an (ordinary) language L, the suffix closure of L is defined by $\mathrm{suffix}(L) = \{s \in A^* \mid \exists u \in A^* \text{ with } us \in L\}$. For partial languages, the suffix closure is defined in the same way as the prefix closure, i.e. componentwise. There is the following relation between the transition structure of L and its suffix closure operator.

Observation 2.4. *For any (partial) language L:* $\mathrm{suffix}(L) = \cup_{u\in L^2} L_u$.

Proof. It is immediate from the fact that $L_u = (\{s \in A^* \mid us \in L^1\},\ \{s \in A^* \mid us \in L^2\})$. □

3 Weak transition structures

In the following definition we introduce the notion of weak derivative (transition). Roughly speaking it disregards unobservable steps, which correspond to so called internal moves in the framework of process algebras [17]. Let $A = A_o \cup A_{uo}$ be a partition of A into observable events (A_o) and unobservable (A_{uo}) events with the natural projection $P : A^* \to A_o^*$. Recall that $P(a) = \varepsilon$ for any $a \in A_{uo}$, $P(a) = a$ for $a \in A_o$, and P is catenative.

Definition 3.1. *(Nondeterministic weak transitions.) For an event $a \in A$ define $L \overset{P(a)}{\Rightarrow}$ if $\exists s \in A^* : P(s) = P(a)$ and $L \overset{s}{\to} L_s$. Denote in this case $L \overset{P(a)}{\Rightarrow} L_s$.*

Remark 3.1. *According to this notation for unobservable events $L \overset{\varepsilon}{\Rightarrow}$ is an abbreviation for $\exists \tau \in A_{uo}^*$ such that $L \overset{\tau}{\to}$. We admit $\tau = \varepsilon$, hence $L \overset{\varepsilon}{\Rightarrow}$ is always true. For $a \in A_o$ our notation*

means that there exist $\tau, \tau' \in A_{uo}^*$ *such that* $L \stackrel{\tau a \tau'}{\to} L_{\tau a \tau'}$. *This definition can be extended to strings (words in* A^**) in the following way:*
$L \stackrel{P(s)}{\Rightarrow}$ *iff* $\exists t \in A^* \ : \ P(s) = P(t)$ *and* $L \stackrel{t}{\to}$. *Denote in this case* $L \stackrel{P(s)}{\Rightarrow} L_t$.

There may exist two or more $s \in A^*$ satisfying the condition in the definition of weak transition. Hence, the weak transition structure introduced above is not deterministic. We introduce deterministic weak transition structure on $\mathcal{L}$ in the following definition.

Definition 3.2. *(Deterministic weak transitions.) Define for* $a \in A_o$: $L \stackrel{a}{\Rightarrow} L_{\hat{a}}$ *if* $L \stackrel{P(a)}{\Rightarrow}$ *and* $L_{\hat{a}} := \cup_{\{s \in L^2 \mid P(s)=a\}} L_s$.

To avoid any confusion we must distinguish between both concepts. Let us introduce the convention that for nondeterministic weak transition we say that $L \stackrel{P(a)}{\Rightarrow} L'$ for some L' and for deterministic concept we denote always the unique weak a-derivative by $L_{\hat{a}}$. For ε-weak transitions we introduce the notation $L \stackrel{\varepsilon}{\Rightarrow} L_{uo}$, where $L_{uo} = \cup\{L_\tau, \ \tau \in A_{uo}^* \text{ such that } L_\tau \text{ exists}\}$, the latter set being nonempty ($\varepsilon \in A_{uo}^*$). Sometimes it will be denoted for notational convenience also by $L_{\hat{\varepsilon}}$, i.e. $L_{\hat{\varepsilon}} = L_{ou}$ is the so called unobservable reach of the partial language L. Notice that for any $L \in \mathcal{L}$, L_{uo} has the pleasant property that for $a \in A_o$: $L_{uo} \stackrel{P(a)}{\Rightarrow}$ iff $L_{uo} \stackrel{a}{\to}$.

The concept of deterministic weak transitions can be extended to observable strings by induction. It should be clear that for $s \in A_o^*$: $L \stackrel{s}{\Rightarrow} L_{\hat{s}}$ iff $L \stackrel{P(s)}{\Rightarrow}$ with $L_{\hat{s}} = \cup_t \{L_t \mid t \in L^2 \text{ and } P(t) = s\}$. Otherwise stated $L_{\hat{s}} = \cup\{L' \mid L \stackrel{P(s)}{\Rightarrow} L'\}$.

There is the following relation between the deterministic weak transitions of a language L and the (strong) transition structure of the projected language $P(L)$ over alphabet A_o.

Proposition 3.2. *For any (partial) language* L *and* $s \in A_o^* : \ P(L) \stackrel{s}{\to} P(L)_s$ *iff* $L \stackrel{s}{\Rightarrow} L_{\hat{s}}$, *in which case* $P(L)_s = P(L_{\hat{s}})$.

Proof. The first part is easy. Indeed, $P(L) \stackrel{s}{\to} P(L)_s$ iff $s = P(s) \in P(L)^2$ iff $\exists u \in L^2 \ : P(u) = P(s)$ iff $L \stackrel{s}{\Rightarrow}$, which is equivalent to $L_{\hat{s}}$ exists, i.e. $L \stackrel{s}{\Rightarrow} L_{\hat{s}}$. In order to see the second part, observe that $t \in P(L_{\hat{s}})$ iff $\exists w \in L_{\hat{s}}$ with $t = P(w)$ iff $\exists w \in L_r$ for $P(r) = s$ and $w \in P^{-1}(t)$ iff $\exists rw \in L$ with $r \in P^{-1}(s)$ and $w \in P^{-1}(t)$ iff $P^{-1}(s)P^{-1}(t) \cap L = P^{-1}(st) \cap L \neq \emptyset$. On the other hand $t \in P(L)_s$ iff $st \in P(L)$ iff $\exists u \in L$: $P(u) = st$ iff $\exists u \in L \cap P^{-1}(st)$ iff $P^{-1}(st) \cap L \neq \emptyset$. Since this is valid for both components of the languages involved, this achieves the proof of the Proposition. □

Notice that deterministic weak derivatives can be generated by strong derivatives and $\varepsilon-$weak derivatives.

Proposition 3.3. *For any partial language* L *and* $a \in A_o$: $L_{\hat{a}} = ((L_{\hat{\varepsilon}})_a)_{\hat{\varepsilon}}$.

Proof. It is immediate from Definition 3.2 and the fact that $P^{(-1)}(a) = \{\tau a \tau' \mid \tau, \tau' \in A_{uo}^*\}$. □

Much more can be said about the topic of weak transition. In particular, our notion of deterministic weak transition gives rise to another concept of weak bisimulation, where usual nondeterministic weak transitions [17] are replaced by the deterministic ones. However, due to Proposition 3.2 it is not difficult to show that the new concept would coincide with the equality of projections, i.e. observable trace equivalence. The notion of weak bisimulation can be defined unlike

deterministic weak transitions for any partial automaton using the concept of powerset automaton introduced in the next subsection. But this goes beyond the aimed scope of the present paper. On the other hand weak bisimulation is not a congruence, but only a weak congruence, and therefore does not provide (strong) quotients. The same problem can be encountered in the framework of process algebras [16].

3.1 Weak transitions and observers for partial automata

Weak transitions in the final automaton of partial languages have been introduced. Let us extend our definition to arbitrary partial automata. As for the nondeterministic concept of weak transitions, the definition is straightforward (simply for a state s in partial automaton $S = (S, \langle o, t\rangle)$ and $a \in A$ we put $s \stackrel{P(a)}{\Rightarrow} s'$ if there exists $u \in A^*$ such that $P(u) = P(a)$ and $s \stackrel{u}{\rightarrow} s' = s_u$). As for the deterministic concept of weak transitions, the corresponding definition does not make a sense in general, however it can be defined if the set of states is a powerset. This motivates the following construction, where we denote the set of nonempty subsets of S by $Pwr^+(S)$ $(= Pwr(S) \setminus \emptyset)$.

Definition 3.3. *(Powerset automaton.) To any partial automaton $S = (S, \langle o, t\rangle)$ we assign a powerset automaton, a partial automaton denoted by $Pwr(S) = (Pwr^+(S), \langle o_S, t_S\rangle)$, where for any $Q \subseteq S;\ Q \neq \emptyset$ we put*

$$t_S(Q)(a) = \cup_{q\in Q}\ t(q)(a) \text{ and } o_S(Q) = \max(o(q), q \in Q).$$

Notice that in the definition of transition function in a powerset automaton there is no necessity to consider separately the case when $t(q)(a)$ is not defined for some $q \in Q$, because according to the definition of partial automata for such a case there is $t(q)(a) = \emptyset \in \Uparrow$. Therefore the above compact way of defining t_S is correct. If we denote by $l_S :\ Pwr^+(S) \rightarrow \mathcal{L}$ and $l :\ S \rightarrow \mathcal{L}$ the unique homomorphisms defined by finality of $\mathcal{L}$, then clearly $l_S(Q) = \cup_{q\in Q}\ l(q)$. This enables us to use the same notation for l and l_S, i.e. the subscript S can be dropped.

In order to implement the projections we define the projected automaton.

Definition 3.4. *(Projected automaton.) The projected automaton is a partial automaton over A_o:*

$$P(S) = (Pwr^+(S), \langle o_P, t_P\rangle) \text{ with}$$

$$t_P(Q)(a) = \cup_{\{w\in A^* |\ P(w)=a\}}\ t_S(Q)(w),\ a \in A_o$$

and $o_P(Q) = o_S(Q_{uo}) = \max\{o(q),\ q \in Q_{uo}\}$, where Q_{uo} is the unobservable reach set of Q, i.e. $Q_{uo} = \{q' \in S \mid \exists q \in Q \text{ with } q \stackrel{\varepsilon}{\Rightarrow} q'.\}$

If we denote by $l_P :\ Pwr^+(S) \rightarrow \mathcal{L}_o$ the unique homomorphism defined by finality of $\mathcal{L}_o$ (automaton of partial languages over A_o), then clearly $l_P(Q) = P(l(Q))$.
Deterministic weak transitions can now be defined in powerset automata: for any $\emptyset \neq Q \subseteq S$ and $a \in A_o$: $Q \stackrel{\hat{a}}{\Rightarrow} Q_{\hat{a}} = \cup_{\{u\in P^{-1}(a)\}}\ t_S(Q)(u)$. Notice in particular that deterministic weak transitions in the powerset automaton $Pwr(S)$ correspond exactly to strong transitions in the projected automaton $P(S)$, i.e. for any $\emptyset \neq Q \subseteq S$: $t_P(Q)(a) = Q_{\hat{a}}$.

The projected automaton $P(S)$ of a given automaton S is related to the observer automaton introduced in [5], but its state space is in general much larger than that of the observer automaton. In control theory, the observer automaton is defined by induction starting from the initial state. However, partial automata as defined above have no initial state. Note that it is natural to consider $L \in \mathcal{L}$ itself as the initial state of the minimal recognizer $\langle L\rangle$ of $L \in \mathcal{L}$. For general S, if we are

given an initial state, the usual construction of observer has its coalgebraic meaning. We define for each partial automaton with designated initial state $S = (S, \langle o, t\rangle)$ with $s_0 \in S$ the observer automaton as the subautomaton of $P(S)$ generated by $\{s_0\}_{uo}$ (accessible from $\{s_0\}_{uo}$):

Definition 3.5. *(Observer automaton.) The observer automaton is denoted by* $Obs(S) = (S_{obs}, \langle o_{obs}, t_{obs}\rangle)$ *with carrier set* $S_{obs} \subseteq Pwr^+(S)$ *defined as follows:*
1) $\{s_0\}_{uo} \in S_{obs}$ *(*$\{s_0\}_{uo}$ *- unobservable reach set is that of* $Pwr(S)$*).*
2) If $Q \in S_{obs}$ *then* $\forall a \in A_o$*:* $t_P(Q)(a) \in S_{obs}$*.*
The structure of the observer automaton is given by the structure of $P(S)$ *restricted to* S_{obs}*:* $\forall Q \in S_{obs}$ *:* $o_{obs}(Q) = o_P(Q)$ *and* $\forall a \in A_o : t_{obs}(Q)(a) = t_P(Q)(a)$*.*

Remark 3.4. *Notice that the definition above implies that the states of the observer are isomorphic to, i.e. can be identified with, different deterministic weak derivatives of its initial state* $\{s_0\}_{uo}$*, i.e. we have* $S_{obs} = \{(s_0)_{\hat{d}} : \ t_P(\{s_0\}_{uo})(d) \text{ is defined.}\}$ *Note that if it happens that two deterministic weak derivatives are equal, they determine a single state of the observer automaton.*

4 Observability relation

In the supervisory control of DES with partial observations the observability of a (specification) language with respect to the plant and projection (to observable events) is necessary for achieving this language as a desirable behavior of the closed-loop system. We assume that $A = A_c \cup A_{uc}$ is a partition of A into controllable events(A_c) and uncontrollable (A_{uc}) events. The observability condition has been first introduced in [14] using a slightly different, but equivalent formulation. This notion of observability is very different from the observability of linear systems. The first attempt to capture this type of condition in an abstract setting goes back to [31]. There has been yet another approach to the observability of DES, based on automata theoretic framework. A necessary condition for a given specification represented by an automaton to be achieved has been formulated in [1] using automata framework.

Definition 4.1. *(Observability.) A partial language* K *is said to be observable with respect to another partial language* L *(with* $K \subseteq L$*) and projection* P *if for all* $s \in K^2$ *and* $a \in A_c$ *the following implication holds true :*

$$sa \in L^2,\ s'a \in K^2, \text{ and } P(s) = P(s') \ \Rightarrow \ sa \in K^2.$$

Our aim is to find a relational characterization of observability. Unlike [9] we present first definitions on automata representations of K and L without specialization to relations on $\mathcal{L}$. The following auxiliary relation is needed.

Definition 4.2. *(Observational indistinguishability relation on* S*.) A binary relation* $Aux(S)$ *on* S*, called* observational indistinguishability relation *is the smallest relation satisfying:*

(i) $\langle s_0, s_0\rangle \in Aux(S)$

(ii) *If* $\langle s, t\rangle \in Aux(S)$ *then* $\forall a \in A$: $(s \overset{P(a)}{\Rightarrow} s'$ *for some* s' *and* $t \overset{P(a)}{\Rightarrow} t'$ *for some* $t')$ $\Rightarrow$ $\langle s', t'\rangle \in Aux(S)$

From the definition of weak transitions it follows that (ii) is equivalent to (ii)' and (iii)' below:
(ii)' If $\langle s, t\rangle \in Aux(S)$ then : ($s \overset{\varepsilon}{\Rightarrow} s'$ for some s' and $t \overset{\varepsilon}{\Rightarrow} t'$ for some t') $\Rightarrow$ $\langle s', t'\rangle \in Aux(S)$
(iii)' If $\langle s, t\rangle \in Aux(S)$ then $\forall a \in A_o$: ($s \overset{a}{\rightarrow} s_a$ and $t \overset{a}{\rightarrow} t_a$) $\Rightarrow$ $\langle s_a, t_a\rangle \in Aux(S)$.
$Aux(S)$ can be characterized by the following lemma.

Lemma 4.1. *For any $s, s' \in S$: $\langle s, s' \rangle \in Aux(S)$ iff there exist two strings $w, w' \in K^2$ such that $P(w) = P(w')$ and $s = (s_0)_w$ and $s' = (s_0)_{w'}$.*

Proof. ($\Leftarrow$) Let $s, s' \in S$ such that there exist two strings $w, w' \in K^2$ such that $P(w) = P(w')$ and $s = (s_0)_w$ and $s' = (s_0)_{w'}$. Let $w = w_1 \dots w_n$ and $w' = t_1 \dots t_m$. Let $P(w) = P(w') = a_1 \dots a_k$. Then $n \geq k$ and $m \geq k$ and there exists two increasing sequences of integers (indices) $u_i \geq i,\ i = 1, \dots, k$ and $v_i \geq i,\ i = 1, \dots, k$ such that $a_i = w_{u_i}$ and $a_i = t_{v_i}$. Since all a_i are observable events we can write $s_0 \overset{P(a_1)\dots P(a_k)}{\Longrightarrow} s$ and $s_0 \overset{P(a_1)\dots P(a_k)}{\Longrightarrow} s'$, whence by (ii) inductively applied $\langle s, s' \rangle \in Aux(S)$.
($\Rightarrow$) Let $\langle s, s' \rangle \in Aux(S)$. Then by the construction of $Aux(S)$ there exist $a_1, \dots, a_k \in A$ such that $s_0 \overset{P(a_1)\dots P(a_k)}{\Longrightarrow} s$ and $s_0 \overset{P(a_1)\dots P(a_k)}{\Longrightarrow} s'$. Therefore there exist by definition of nondeterministic weak transitions two strings w, w' with the same projection such that $s = (s_0)_w$ and $s' = (s_0)_{w'}$. □

Remark 4.2. *Remark that $Aux(S)$ is not in general an equivalence relation, because it might be non transitive as is shown in the following example. However it is always symmetric and reflexive. Such a relation is sometimes called a tolerance relation.*

Example 1. *Take $S = DK$, where $K = (\emptyset, \{\overline{a\tau}, \tau\})$. Then $DK = \{K, K_a, K_\tau\}$. Then $Aux(DK)$ is not an equivalence relation, because $K_\tau = K_{a\tau} = \{\varepsilon\}$ means that $\langle K, K_\tau \rangle \in Aux(DK)$ and $\langle K_\tau, K_a \rangle \in Aux(DK)$, while $\langle K, K_a \rangle \notin Aux(DK)$.*

A natural question arises under which conditions $Aux(S)$ is an equivalence relation. Recall the concept of state-partition automaton from [6] and [5].

Definition 4.3. *(State-partition automaton.) Let $S = (S, \langle o, t \rangle)$ be a partial automaton and let $Obs(S) = (S_{obs}, \langle o_{obs}, t_{obs} \rangle)$ be its observer automaton. Then S is said to be a state-partition automaton if for all $Q_1, Q_2 \in S_{obs} \subseteq Pwr(S)$ we have: $Q_1 \neq Q_2 \ \Rightarrow\ Q_1 \cap Q_2 = \emptyset$.*

A partial automaton S with initial state s_0 is a state-partition automaton if any two different states of the observer are disjoint (as subsets of S). In the case, where all states of the automaton S are accessible from s_0, this condition is equivalent to the statement that the states of the observer automaton form a partition of S. In our coalgebraic framework, the property of state-partition automaton can be described in terms of deterministic weak derivatives (in the sense of $Pwr(S)$). Namely, S is a state-partition automaton if $\forall d, d' \in P(l(s_0)^2)$: $(s_0)_{\hat{d}} \neq (s_0)_{\hat{d'}} \quad \Rightarrow \quad (s_0)_{\hat{d}} \cap (s_0)_{\hat{d'}} = \emptyset$. It is easy to prove that this condition is sufficient for $Aux(S)$ to be an equivalence relation.

Proposition 4.3. *If S is a state-partition automaton then $Aux(S)$ is an equivalence relation.*

Proof. Let S be a state-partition automaton. Let us show that $Aux(S)$ is transitive. Take s, s', s'' such that $\langle s, s' \rangle \in Aux(S)$ and $\langle s', s'' \rangle \in Aux(S)$. Let us show that $\langle s, s'' \rangle \in Aux(S)$. There exist strings v, v', w, w' such that $P(v) = P(v')$, $P(w) = P(w')$, $s = (s_0)_v$, $s' = (s_0)_{v'}$, $s' = (s_0)_{w'}$, and $s''(s_0)_w$. Denote $d = P(v)$ and $d' = P(w)$. Then $s' \in (s_0)_{\hat{d}} \cap (s_0)_{\hat{d'}}$. This means that $(s_0)_{\hat{d}} \cap (s_0)_{\hat{d'}} \neq \emptyset$ and by definition of state-partition automaton $(s_0)_{\hat{d}} = (s_0)_{\hat{d'}}$. In particular, there exists w'' with $P(w'') = P(w)$ and $s = (s_0)_{w''}$. Thus $\langle s, s'' \rangle \in Aux(S)$. □

However the opposite statement does not hold as the following example shows.

Example 2. *Put $S = \{s_0, s_1, s_2\}$ with the transition function $t(s_0)(a) = s_2,\ t(s_0)(\tau) = s_1,\ t(s_1)(\tau) = s_2,\ t(s_2)(\tau) = s_1$, the other transitions are undefined, and the output function can be arbitrary. Then $Aux(S) = S^2$ is trivially an equivalence relation on S, but S is not a state partition automaton, because the sets $(s_0)_{uo} = S$ and $(s_0)_{\hat{a}} = \{s_1, s_2\}$ violate the condition for S to be a state-partition automaton.*

Lemma 4.1 implies that $\langle s, s' \rangle \in Aux(S)$ iff there exists $d \in P(L)$ $(d = P(w) = P(w'))$ such that $s \in (s_0)_{\hat{d}}$ and $s' \in (s_0)_{\hat{d}}$, i.e. there exists a state $Q \subseteq S$ of the observer automaton $Obs(S)$ such that $s \in Q$ and $s' \in Q$. This motivates the following definition.

Definition 4.4. *Define for any $s \in S$:*
$\lfloor s \rfloor_{Aux(S)} = \{s' \in S : \langle s, s' \rangle \in Aux(S)\}$.

Let us now use a simpler notation $\lfloor s \rfloor_{Aux}$ if automaton S is supposed to be fixed. It is to be expected that

Observation 4.4. *The following properties are equivalent:*

(i) *$Aux(S)$ is an equivalence relation*

(ii) $\forall s, s' \in S : \langle s, s' \rangle \in Aux(S) \ \Rightarrow \ \lfloor s \rfloor_{Aux} = \lfloor s' \rfloor_{Aux}$.

(iii) $\forall s, s' \in S : \lfloor s \rfloor_{Aux} \cap \lfloor s' \rfloor_{Aux} \neq \emptyset \ \Rightarrow \ \lfloor s \rfloor_{Aux} = \lfloor s' \rfloor_{Aux}$.

Proof. $(i) \ \Rightarrow (ii)$ Let $Aux(S)$ be an equivalence relation and take arbitrary $s, s' \in S$ such that $\lfloor s \rfloor_{Aux} \neq \lfloor s' \rfloor_{Aux}$. Assume that $\exists q \in S : q \in \lfloor s \rfloor_{Aux} \setminus \lfloor s' \rfloor_{Aux}$, the other case can be treated in a symmetric way. By definition of $\lfloor . \rfloor_{Aux}$, $\langle s, q \rangle \in Aux(S)$ and $\langle q, s' \rangle \notin Aux(S)$. Let us show that $\langle s, s' \rangle \notin Aux(S)$. Suppose by contradiction that $\langle s, s' \rangle \in Aux(S)$, then using the fact that $Aux(S)$ is symmetric and transitive, $\langle s', q \rangle \in Aux(S)$, hence also $\langle q, s' \rangle \in Aux(S)$, a contradiction. Therefore $\langle s, s' \rangle \notin Aux(S)$.
$(ii) \ \Rightarrow (iii)$ Let the implication (ii) hold true. We show (iii): if for $s, s' \in S : \ \lfloor s \rfloor_{Aux} \cap \lfloor s' \rfloor_{Aux} \neq \emptyset$, then there exists $q \in S$ such that $q \in \lfloor s \rfloor_{Aux} \cap \lfloor s' \rfloor_{Aux}$, i.e. $\langle s, q \rangle \in Aux(S)$ and $\langle q, s' \rangle \in Aux(S)$. Thus from (ii) we have $\lfloor s \rfloor_{Aux} = \lfloor q \rfloor_{Aux} = \lfloor s' \rfloor_{Aux}$, which was to be shown.
$(iii) \ \Rightarrow (i)$ Let the implication (iii) hold true. Take $\langle s, s' \rangle \in Aux(S)$, and $\langle s', s'' \rangle \in Aux(S)$. Then $s' \in \lfloor s \rfloor_{Aux} \cap \lfloor s'' \rfloor_{Aux} \neq \emptyset$. It follows that $\lfloor s \rfloor_{Aux} = \lfloor s'' \rfloor_{Aux}$. But $s \in \lfloor s \rfloor_{Aux} = \lfloor s'' \rfloor_{Aux}$, i.e. $\langle s, s'' \rangle \in Aux(S)$ according to the definition of $\lfloor s'' \rfloor_{Aux}$. This proves the transitivity of $Aux(S)$. □

Note that in fact $\lfloor s \rfloor_{Aux} = \cup_{\{d:\ s \in (s_0)_{\hat{d}}\}} (s_0)_{\hat{d}}$. It follows that $\langle s, s' \rangle \in Aux(S)$ iff $\{s, s'\} \subseteq Q$, for some $Q \in S_{obs}$, which is equivalent by definition of the observer to $\exists d \in A_o^* : \ \{s, s'\} \subseteq (s_0)_{\hat{d}}$.

One could ask for conditions that ensure that $Aux(S)$ is an equivalence relation without the use of $Aux(S)$ itself. Let $M = l(s_0)^2$ be the closed behavior generated by s_0. There is the following condition using the states of $Obs(S)$:

Lemma 4.5. *$Aux(S)$ is an equivalence relation iff* $\forall d_1, d_2 \in P(M) : \ (s_0)_{\hat{d_1}} \cap (s_0)_{\hat{d_2}} \neq \emptyset \ \Rightarrow (\forall s_1 \in (s_0)_{\hat{d_1}}$ *and* $\forall s_2 \in (s_0)_{\hat{d_2}}) \ \exists d \in P(M) : \ \{s_1, s_2\} \subseteq (s_0)_{\hat{d}}$.

Proof. $(\Rightarrow)$ Let $Aux(S)$ be an equivalence relation and suppose by contradiction that $\exists d_1, d_2 \in P(M)$: $(s_0)_{\hat{d_1}} \cap (s_0)_{\hat{d_1}} \neq \emptyset$ and $\exists s_1 \in (s_0)_{\hat{d_1}}$ and $\exists s_2 \in (s_0)_{\hat{d_2}}$): $\forall d \in P(M) : \ \{s_1, s_2\} \not\subseteq (s_0)_{\hat{d}}$. It means that there exists $q \in S$: $q = (s_0)_{v_1} = (s_0)_{v_2}$, where $P(v_1) = d_1$ and $P(v_2) = d_2$, i.e. $\langle s_1, q \rangle \in Aux(S)$ and $\langle q, s_2 \rangle \in Aux(S)$. By the transitivity of $Aux(S)$, $\langle s_1, q \rangle \in Aux(S)$ and

$\langle q, s_2\rangle \in Aux(S)$ implies $\langle s_1, s_2\rangle \in Aux(S)$, i.e. there exists $d \in P(M)$: $s_1 \in (s_0)_{\hat{d}}$ and $s_2 \in (s_0)_{\hat{d}}$, a contradiction with $\forall d \in P(M) : \{s_1, s_2\} \not\subseteq (s_0)_{\hat{d}}$. Thus the implication holds true. ($\Leftarrow$) Assume the implication on the right hand side holds true, $\langle s, s'\rangle \in Aux(S)$, and $\langle s', s''\rangle \in Aux(S)$. By Lemma 4.1 there exists strings $v, v', w, w' \in M$ such that $s = (s_0)_v$, $s' = (s_0)_{v'} = (s_0)_{w'}$, $s'' = (s_0)_w$, where $P(v) = P(v')$, and $P(w) = P(w')$. Denote by $d_1 = P(v)$ and $d_2 = P(w)$. Then $s' \in (s_0)_{\hat{d_1}} \cap (s_0)_{\hat{d_2}}$. Therefore for $s \in (s_0)_{\hat{d_1}}$ and $s'' \in (s_0)_{\hat{d_2}}$ there exists $d \in P(M)$: $s \in (s_0)_{\hat{d}}$ and $s'' \in (s_0)_{\hat{d}}$, i.e. $\langle s, s''\rangle \in Aux(S)$ using Lemma 4.1, and $Aux(S)$ is an equivalence relation. □

Notice that the condition of this Lemma is similar but somewhat weaker than the condition required for S to be a state-partition automaton, which is only a sufficient condition for $Aux(S)$ to be an equivalence relation.

Our aim now is to provide a coalgebraic characterization of observability. Since observability is a property of the second (closed) components of K and L, we can assume that $S_1 = (S_1, \langle o_1, t_1\rangle)$ is a partial automaton with initial state $s_0 \in S$ that represents K in the sense $K = l_1(s_0)$, $l_1 : S_1 \to \mathcal{L}$ being the unique behavior homomorphism defined by finality of $\mathcal{L}$. Moreover, since $K \subseteq L$, we can assume that S_1 is a subautomaton of $S = (S, \langle o, t\rangle)$ with $L = l(s_0)$ ($l : S \to \mathcal{L}$ is the behavior homomorphism) and s_0 their common initial state. Let the transition function of S be denoted by $\to$, i.e. $s \xrightarrow{a} s_a$ means $s_a = t(s)(a)$ and similarly the transition function t_1 of S_1 is denoted by $\to_1$, i.e. $s \xrightarrow{a}_1 s^1_a$ means $s^1_a = t_1(s)(a)$. Notice also that due to the requirement that S_1 is a subautomaton of S, we have in fact $s^1_a = s_a \in S_1$. It means that the superscript ^{1} can be dropped here. Let us introduce observability relations, in which the observational indistinguishability relation is involved.

Definition 4.5. *(Observability relation.) A binary relation $O(S_1, S)$ on $S_1 \times S$ is called the* observability relation *if for any $\langle s, t\rangle \in O(S_1, S)$ the following items hold:*

(i) $\forall a \in A : s \xrightarrow{a}_1 s_a \;\Rightarrow\; t \xrightarrow{a} t_a$ *and* $\langle s_a, t_a\rangle \in O(S_1, S)$

(ii) $\forall a \in A_c : t \xrightarrow{a} t_a$ *and* $(\exists s' : \langle s, s'\rangle \in Aux(S_1) : s' \xrightarrow{a}_1 s'_a) \;\Rightarrow\; s \xrightarrow{a}_1 s_a$ *and* $\langle s_a, t_a\rangle \in O(S_1, S)$.

Remark that (ii) can be expressed using the set $\lfloor s\rfloor_{Aux(S_1)}$ introduced above: the condition $\exists s' : \langle s, s'\rangle \in Aux(S_1) : s' \xrightarrow{a}_1 s'_a$ can be replaced by the simpler one $\lfloor s\rfloor_{Aux(S_1)} \xrightarrow{a}_1$, where $\to_1$ is now to be interpreted in $Pwr(S_1)$. For $s \in S_1$ and $s' \in S$ we write $s \approx_{O(S_1,S)} s'$ whenever there exists an observability relation $O(S_1, S)$ on $S_1 \times S$ such that $\langle s, s'\rangle \in O(S_1, S)$. Now we are ready to prove:

Theorem 4.6. *A (partial) language K is observable with respect to L ($K \subseteq L$) and P iff $s_0 \approx_{O(S_1,S)} s_0$.*

Proof. ($\Rightarrow$) Let K be observable with respect to L and P such that $K \subseteq L$. Denote

$$O(S_1, S) = \{\langle (s_0)_u, (s_0)_u\rangle \in S_1 \times S \mid u \in K^2\} \subseteq S_1 \times S.$$

Note that some of pairs in $O(S_1, S)$ can be equal. Indeed, it is possible that there exists $u, v \in K^2 : (s_0)_u = (s_0)_v$. But we will show that this is not a problem for our proof. Let us show that $O(S_1, S)$ is an observability relation. Let $\langle q, r\rangle \in O(S_1, S)$. Because of the definition of $O(S_1, S)$ we can assume that $q = (s_0)_s$ for some $s \in K^2$ and $r = q$. We must show that conditions (i) and (ii) are satisfied.

(i) Let $q \xrightarrow{a}_1$ for $a \in A$. Clearly $q \xrightarrow{a}$, because S_1 is a subautomaton of S and it is immediate from

the definition of $O(S_1, S)$ that $\langle q_a, q_a \rangle \in O(S_1, S)$.
(ii) Let $q \stackrel{a}{\rightarrow}$ for $a \in A_c$ and $\exists q' : \langle q, q' \rangle \in Aux(S_1) : \quad q' \stackrel{a}{\rightarrow}_1$. Then by Lemma 4.1 there exist two strings $s', s'' \in K^2$ such that $P(s'') = P(s')$, $q = (s_0)_{s''}$, and $q' = (s_0)_{s'}$. Now $q' \stackrel{a}{\rightarrow}_1$ implies that $s'a \in K^2$. Recall that $s' \in K^2$, because $q' = (s_0)_{s'}$. From $q \stackrel{a}{\rightarrow}$ and $q = (s_0)_{s''}$ follows $s''a \in L^2$ and by application of the observability of K with respect to L and P we deduce $s''a \in K^2$, i.e. $a \in l\left((s_0)_{s''}\right)^2$, which means that $q = (s_0)_{s''} \stackrel{a}{\rightarrow}_1$. It follows from (i) that $\langle q_a, q_a \rangle \in O(S_1, S)$. We see now that considering s'' instead of s, where $q = (s_0)_s = (s_0)_{s''}$ did not make any difference.
($\Leftarrow$) Let $s_0 \approx_{O(S_1,S)} s_0$. Let us show that K is observable with respect to L and P. For this purpose, let $s \in K^2$, $s'a \in K^2$ for $a \in A_c$, $sa \in L^2$, and $P(s) = P(s')$. Then $s \in K^2 \cap L^2$, i.e. $(s_0) \stackrel{s}{\rightarrow}_1$ and $(s_0) \stackrel{s}{\rightarrow}$, whence from (i) of Definition 4.5 inductively applied $(s_0)_s \approx_{O(S_1,S)} (s_0)_s$. Since K^2 is prefix closed, $s' \in K^2 = l_1(s_0)^2$, we have $s_0 \stackrel{s'}{\rightarrow}_1 (s_0)_{s'}$ and according to Lemma 4.1 we have $\langle (s_0)_s, (s_0)_{s'} \rangle \in Aux(S_1)$. Now we have $(s_0)_s \stackrel{a}{\rightarrow}$ and $(s_0)_{s'} \stackrel{a}{\rightarrow}_1$, where recall $\langle (s_0)_s, (s_0)_{s'} \rangle \in Aux(S_1)$. By (ii) of the definition of observability relation we obtain that $(s_0)_s \stackrel{a}{\rightarrow}_1$, i.e. $(s_0) \stackrel{sa}{\rightarrow}_1$, which means that $sa \in l_1(s_0)^2 = K^2$. □

Recall that the tests for observability proposed in [7] or [5] are to be made for all states of observer automaton $Obs(S_1)$ that are in fact different weak derivatives $(s_0)_{\hat{d}},\ d \in P(l_1(s_0)^2)$. It would mean that the test for observability requires the construction of the observer. We have just shown that a test for observability might not rely on the observer, but on $Aux(S_1)$ instead. The test for condition (ii) of observability relation can be made for different $\lfloor s \rfloor_{Aux(S_1)}$. Recall that $\lfloor s \rfloor_{Aux(S)} = \cup_{\{d:\, s \in (s_0)_{\hat{d}}\}} (s_0)_{\hat{d}}$.

5 Normal relations

In this section we show an application of the above introduced notion of weak transition to the characterization of normality of languages introduced in supervisory control of DES with partial observations. For the sake of completeness, the concept of normality ([14], [4], [6], etc.) is stated.

Definition 5.1. *(Normality.) Let $K, L \in \mathcal{L}$: $K \subseteq L$. K is said to be (L, P)-normal if $K^2 = L^2 \cap P^{-1}(P(K^2))$.*

Property 5.1. *K is (L, P)-normal iff $s \in K^2,\ s' \in L^2$, and $P(s) = P(s') \quad \Rightarrow \quad s' \in K^2$.*

Proof. Since $K \subseteq L$, normality is equivalent to $L^2 \cap P^{-1}(P(K^2)) \subseteq K^2$, which is equivalent to the statement. □

From the definitions of strong and weak transitions it follows:

Corollary 5.2. *$K \subseteq L$ is (L, P)-normal iff $\forall w \in A^*$: ($L \stackrel{w}{\rightarrow}$ and $K \stackrel{P(w)}{\Rightarrow}$) $\quad \Rightarrow \quad K \stackrel{w}{\rightarrow}$.*

Normality is preserved by the unobservable reach sets.

Proposition 5.3. *If a language K is $(L, P)-$normal then K_{uo} is $(L_{uo}, P)-$normal.*

Proof. Using Corollary 5.2 it is sufficient to show that $\forall w \in A^*$: $L_{uo} \stackrel{w}{\rightarrow}$ and $K_{uo} \stackrel{P(w)}{\Rightarrow}$ implies that $K_{uo} \stackrel{w}{\rightarrow}$. Recall that $L_{uo} = \cup\{L_\tau \mid \tau \in L^2 \text{ and } P(\tau) = \varepsilon\}$. Assume that $L_{uo} \stackrel{w}{\rightarrow}$ and $K_{uo} \stackrel{P(w)}{\Rightarrow}$, i.e. there exist $\tau, \tau' \in A_{uo}^*$ such that $L_\tau \stackrel{w}{\rightarrow}$ and $K_{\tau'} \stackrel{P(w)}{\Rightarrow}$. Hence, $L \stackrel{\tau w}{\longrightarrow}$ and $K \stackrel{P(\tau' w)}{\Longrightarrow}$, thus $K \stackrel{P(\tau w)}{\Longrightarrow}$,

because $P(\tau' w) = P(\tau w)$. It follows that $K \xrightarrow{\tau w}$ after the application of $(L, P)-$normality of K. But it means that $K_\tau \xrightarrow{w}$. Since $K_\tau \subseteq K_{uo}$ we obtain finally that $K_{uo} \xrightarrow{w}$, which was to be shown. □

The following fact will be useful.

Property 5.4. *Normality is preserved by (strong) transitions, i.e. if K is (L, P)-normal and $a \in A$ such that $K \xrightarrow{a}$ and $L \xrightarrow{a}$ then K_a is (L_a, P)-normal.*

Proof. It is easily seen from Corollary 5.2. Indeed, if $L_a \xrightarrow{w}$ then $L \xrightarrow{aw}$ and if $K_a \overset{P(w)}{\Rightarrow}$ then clearly $K \overset{P(aw)}{\Rightarrow}$, whence by $(L, P)-$normality of K we deduce that $K \xrightarrow{aw}$, i.e. $K_a \xrightarrow{w}$. □

Remark 5.5. *It is interesting to notice that (L, P)- normality is preserved by deterministic weak transitions. It follows from Property 5.4, Proposition 5.3 and Proposition 3.3.*

Now we introduce a binary relation that corresponds to the normality.

Definition 5.2. *(Normal relation.) Given two (partial) automata $S_1 = (S_1, \langle o_1, t_1\rangle)$ and $S = (S, \langle o, t\rangle)$ as in Section 4 with initial state $s_0 \in S$, a binary relation $N(S_1, S)$ on $S_1 \times S$ is called a* normal relation *if for any $\langle s, t\rangle \in N(S_1, S)$ the following items hold:*

(i) $\forall a \in A \ : \ s \xrightarrow{a}_1 s_a \ \Rightarrow \ t \xrightarrow{a} t_a$ *and* $\langle s_a, t_a\rangle \in N(S_1, S)$

(ii) $\forall a \in A_o : t \xrightarrow{a} t_a$ *and* $(\exists s' : \langle s, s'\rangle \in Aux(S_1) : s' \xrightarrow{a}_1 s'_a) \ \Rightarrow \ s \xrightarrow{a}_1 s_a$.

(iii) $\forall u \in A_{uo} : \ t \xrightarrow{u} t_u \ \Rightarrow \ s \xrightarrow{u}_1 s_u$.

Remark 5.6. *Recall that (ii) can be expressed using the set $\lfloor s \rfloor_{Aux(S_1)}$: the condition $\exists s' : \langle s, s'\rangle \in Aux(S_1) : s' \xrightarrow{a}_1 s'_a$ can be replaced by the simpler one $\lfloor s \rfloor_{Aux(S_1)} \xrightarrow{a}_1$.*

For $s \in S_1$ and $t \in S$ we write $s \approx_{N(S_1,S)} t$ whenever there exists a normal relation $N(S_1, S)$ on $S_1 \times S$ such that $\langle s, t\rangle \in N(S_1, S)$. Now we can prove:

Theorem 5.7. *A (partial) language K is $(L, P)-$normal iff $s_0 \approx_{N(S_1,S)} s_0$.*

Proof. $(\Rightarrow)$ Let K be $(L, P)-$normal. Denote

$$R = \{\langle (s_0)_u, (s_0)_u\rangle \mid u \in K^2 \} \subseteq S_1 \times S.$$

Let us show that R is indeed a normality relation. Assume that $\langle q, r\rangle \in R$. From the form of R it follows that we can assume that $q = r = (s_0)_s$ for some $s \in K^2$. The same remark as in the proof of Theorem 4.6 applies. Namely, $s \in K^2$ such that $q = r = (s_0)_s$ might not be uniquely determined. Again we can show that the choice is not important. Nevertheless, since the argument is the same as in the proof of Theorem 4.6, we assume that s has been chosen in the way Lemma 4.1 in (ii) below can be correctly applied.

(i) This part is the same as above in the proof of Theorem 4.6.

(ii) Let $a \in A_o$ be such that $r = q \xrightarrow{a}$ and $\exists q' : \ \langle q, q'\rangle \in Aux(S_1)$ with $q' \xrightarrow{a}_1$. Then by Lemma 4.1 there exists a string $s' \in K^2 = l_1(s_0)^2$ such that $P(s) = P(s')$ and $q' = (s_0)_{s'}$. Thus we have $(s_0)_{s'} \xrightarrow{a}_1$, whence $s'a \in K^2$ and by normality we deduce $sa \in K^2$, because also $sa \in L^2 = l(s_0)^2$ (from $q = (s_0)_s \xrightarrow{a}$).

(iii) Let $u \in A_{uo}$ and $q = (s_0)_s \xrightarrow{u}$. Thus we have and $su \in l(s_0)^2 = L^2$, where $P(su) = P(s)$, and recall that $s \in K^2$, whence by normality (Property 5.1) $su \in K^2$, i.e. $q = (s_0)_s \xrightarrow{u}_1$.

($\Leftarrow$) Now let R be a normal relation on $S_1 \times S$ and $\langle s_0, s_0\rangle \in R$. It will be shown that K is $(L,P)-$normal using Corollary 5.2. Let us prove by induction on the structural complexity of strings that for each $w \in A^* : L \stackrel{w}{\rightarrow}$ and $K \stackrel{P(w)}{\Rightarrow}$ implies $K \stackrel{w}{\rightarrow}$. For $w = \varepsilon$ it is trivially true, because K^2 is prefix closed (i.e. $K \stackrel{\varepsilon}{\rightarrow}$). Suppose now that for $w \in A^*$ the above implication holds true. Let $L \stackrel{wa}{\rightarrow}$ and $K \stackrel{P(wa)}{\Rightarrow}$. This implies in particular that $L \stackrel{w}{\rightarrow}$ and $K \stackrel{P(w)}{\Rightarrow}$, hence by the induction hypothesis $K \stackrel{w}{\rightarrow}$. Since $\langle s_0, s_0\rangle \in R$, by inductive application of (i) of the definition of normal relation we obtain that $\langle (s_0)_w, (s_0)_w\rangle \in R$. Now suppose first $a \in A_o$ and $L_w \stackrel{a}{\rightarrow}$ and $K \stackrel{P(wa)}{\Rightarrow}$. The latter means by definition of nondeterministic weak transition that there exists $v \in A^* : P(v) = P(w)$ and $K \stackrel{v}{\rightarrow} K' \stackrel{P(a)}{\Rightarrow}$, where $K' = K_v$. Moreover, v (and K') can be chosen such that $K' \stackrel{a}{\rightarrow}$. Indeed, $K' \stackrel{P(a)}{\Rightarrow}$ means by definition there exists $\tau, \tau' \in A_{uo}^*$ such that $K' \stackrel{\tau a \tau'}{\rightarrow}$. Thus, it is sufficient to consider $K'' = K'_\tau$ and $v\tau$ rather then v, because now $K'' = K'_\tau \stackrel{a}{\rightarrow}$. But $P(v\tau) = P(v) = P(w)$. Now $L_w \stackrel{a}{\rightarrow}$ gives $(s_0)_w \stackrel{a}{\rightarrow}$ and $v \in K^2$ with $K_v \stackrel{a}{\rightarrow}$ implies $va \in K^2$, i.e. $(s_0)_v \stackrel{a}{\rightarrow}_1$. By Lemma 4.1 $\langle (s_0)_w, (s_0)_v\rangle \in Aux(S_1)$, i.e. by application of (ii) of normal relation we obtain $(s_0)_w \stackrel{a}{\rightarrow}_1$, i.e. $K \stackrel{wa}{\rightarrow}$. In the case $a \in A_{uo}$ the property (iii) gives the same result. Indeed, we simply obtain that $L_w \stackrel{a}{\rightarrow}$, i.e. $(s_0)_w \stackrel{a}{\rightarrow}$ implies by (iii) of the definition of normal relations that $(s_0)_w \stackrel{a}{\rightarrow}_1$. But this means that $K_w \stackrel{a}{\rightarrow}$. □

Let us recall here the concept of control relation introduced in [20]. Let A_{uc} be the subset of uncontrollable events. We use the following stronger version of control relations with condition (i) strengthened to inclusion.

Definition 5.3. *(Control relation.) Given two partial automata $S_1 = (S_1, \langle o_1, t_1\rangle)$ and $S = (S, \langle o, t\rangle)$ as above, a binary relation C on $S_1 \times S$ is called a* control relation *if for any $\langle s, t\rangle \in C$ the following items hold:*

(i) $\forall a \in A : s \stackrel{a}{\rightarrow}_1 s_a \;\Rightarrow\; t \stackrel{a}{\rightarrow} t_a$ *and* $\langle s_a, t_a\rangle \in C$

(ii) $\forall u \in A_{uc} : t \stackrel{u}{\rightarrow} t_u \;\Rightarrow\; s \stackrel{u}{\rightarrow}_1 s_u$ *and* $\langle s_u, t_u\rangle \in C$.

It has been shown in [20] that

Theorem 5.8. *A (partial) language K is controllable with respect to L and A_{uc} iff there exists a control relation $R \subseteq S_1 \times S$ such that $\langle s_0, s_0\rangle \in R$.*

In the above definition, S_1 corresponds to the closed-loop system consisting of the plant and the supervisor and S corresponds to the open-loop plant. From Theorem 4.6 and Theorem 5.7 after comparing the definitions of observability and normal relations it follows immediately the well known fact that normality implies observability. More precisely:

Corollary 5.9. *K is $(L,P)-$normal iff K is observable with respect to L and P and controllable with respect to L and A_{uo}. In particular, we obtain the following well known implication. If K is observable with respect to L and P, controllable with respect to L and A_{uc}, and $A_c \subseteq A_o$ (i.e. $A_{uo} \subseteq A_{uc}$), then K is (L,P)-normal.*

Finally let us compare our result with that in [7]. The test for normality in [7] is made for all classes introduced in that paper that are in fact different weak derivatives $(s_0)_{\hat{d}},\ d \in P(K)$. It means that their test for observability and/or normality requires the construction of an observer. It is known [30] that these tests can be done in polynomial time. We have shown that these tests do not rely on the observer, but on $Aux(S_1)$ instead in exactly the same way as the test for observability.

6 Supremal normal and controllable sublanguages

It is well known that the supremal observable sublanguage of a given language does not always exist. On the other hand, supremal normal and therefore also supremal controllable and normal sublanguages of a given language do exist. Algorithms for their computation have been presented in [6], [7], and [4]. The algorithm in [7] has been developed using the invariance properties of equivalence relations induced on a given language by a natural projection. The concept of active event set of a given state or a subset of states (corresponding to an equivalence class) is used. However, this concept can be captured in a more natural way using the coalgebraic structure of partial automata.

6.1 Construction of supremal normal and controllable sublanguages using normal relations

Given two (ordinary and not necessarily prefix closed) languages K and L such that $K \subseteq L$, let us consider partial automata $S' = (S', \langle o', t' \rangle)$ and $S = (S, \langle o, t \rangle)$ representing K and L in the sense made precise below and such that S' is a subautomaton of S with s_0 their common initial state. Let $l' : S' \to \mathcal{L}$ and $l : S \to \mathcal{L}$ be the associated behavior homomorphisms, where $K = (l'(s_0))^1$ and $L = (l(s_0))^1$. Recall that such a representation with subautomaton always exists [6]. Let the transition function of S be denoted by $\to$, i.e. $s \xrightarrow{a} s_a$ means $s_a = t(s)(a)$ and similarly the transition function t' of S' is denoted by $\to'$, i.e. $s \xrightarrow{a}{}' s_a$ means $s_a = t'(s)(a)$. This notation is possible, because S' is a subautomaton of S. Moreover we can assume without loss of generality that S' is a trim automaton.

We consider normal relations on $S' \times S$. Theorem 5.7 suggests a test for normality. We start with including $\langle s_0, s_0 \rangle \in N(S', S)$ and we continue by adding new states using (i) of the definition of normal relation. Every time a new state is included we test conditions (ii) and (iii), either these conditions are satisfied and we continue the construction of $N(S', S)$ or one of them is not satisfied, in which case the procedure aborts and the conclusion is that K is not $(L, P)-$normal. It is obvious that if this procedure is never aborted, it leads to the diagonal relation on S', denoted by $diag(S')$, because S' is a trim subautomaton of S. In this case $diag(S')$ is a normal relation on $S' \times S$ proving that K is (L, P)-normal. Thus diagonal normal relations are of special interest for testing the normality of a language and computing supremal normal sublanguages. Conditions (ii) and (iii) of normal relations can be reformulated:

(ii) $\forall s \in S' \subseteq S$ and $\forall a \in A_o$: $(s \xrightarrow{a} s_a$ and $\lfloor s \rfloor_{Aux(S')} \xrightarrow{a}{}')$ $\Rightarrow$ $s \xrightarrow{a}{}' s_a$.
(iii) $\forall s \in S' \subseteq S$ and $\forall u \in A_{uo}$: $s \xrightarrow{u} s_u$ $\Rightarrow$ $s \xrightarrow{u}{}' s_u$.

The procedure for computation of the supremal normal sublanguage can now be easily devised. It will consist in removing some strings that cause the violation of conditions (ii) and (iii) above. This amounts to removing some states and edges from automaton S', which is made by Algorithms 1 and 2.

Note that condition (iii) of the definition of normal relation is just the "controllability" condition with respect to A_{uo} instead of A_{uc} that appears in the definition of control relation [20]. Therefore, the first step (algorithm) of our computation will be similar to the case of complete observations, i.e. the removal of certain states that violate our "extended controllability" condition (iii).

Now let us devise an algorithm that ensures condition (iii) of normality, which is of the same

kind as (ii) of control relations. Therefore for computation of supremal normal and controllable sublanguages we can take care of (ii) of controllability and (iii) of normal relations at the same time and use a natural extension of the algorithm presented in [20].

Let $R = \{\langle (s_0)_w, (s_0)_w \rangle \mid w \in K \subseteq L\} \subseteq S' \times S$. Clearly R is a diagonal relation on S', i.e. $R = diag(S')$ and it carries an automaton structure as given in [20]. Recall that for $\langle t, t \rangle \in R$ we put $\langle t, t \rangle \stackrel{a}{\rightarrow}_R \langle t, t \rangle_a$ iff $t \stackrel{a}{\rightarrow}' t_a$ in S', which implies $t \stackrel{a}{\rightarrow} t_a$ (because S' is a subautomaton of S), in which case $\langle t, t \rangle_a = \langle t_a, t_a \rangle$. The output function plays no role here, it can be arbitrary. Now we can repeat the algorithm from [20] for R.

Algorithm 1. *Define the following operator that maps any diagonal relation $H \subseteq S' \times S$ to*

$$\Gamma(H) = \{\langle s, s \rangle \in H \mid \forall u \in A_{uc} \cup A_{uo} : \ s \stackrel{u}{\rightarrow} \Rightarrow \ s \stackrel{u}{\rightarrow}' \ \textit{and} \ \langle s_u, s_u \rangle \in H).\}$$

We put $\hat{R} = \bigcap_{i \geq 0} \Gamma^i(R)$.

Then $\hat{R} = diag(S_1)$ for some $S_1 \subseteq S'$ and $\hat{R}$ is the greatest fixed point of Γ that is contained in R :

Lemma 6.1. *Let Γ and $\hat{R}$ are as defined above. Then 1. $\hat{R} = \Gamma(\hat{R})$ and*
2. For any $R' \subseteq R$: If $R' \subseteq \Gamma(R')$ then $R' \subseteq \hat{R}$.

Proof. The operator Γ is monotone (as a set operator with respect to inclusion), i.e. there exists a fixpoint according to [26]. A detailed proof can be found in [20]. □

The construction in Algorithm 1 yields the supremal sublanguage of K that is controllable with respect to L and $A_{uc} \cup A_{uo}$. More precisely, we have the following theorem:

Theorem 6.2. *Let $K = l'(s_0)^1$, $L = l(s_0)^1$, and automata R and $\hat{R}$ be as above. Then $\langle s_0, s_0 \rangle \in \hat{R}$ and $\hat{l}(\langle s_0, s_0 \rangle)^1 = E$, where $\hat{l} : \hat{R} \rightarrow \mathcal{L}$ is the unique homomorphism describing the behavior of states in $\hat{R}$ and E is the supremal sublanguage of K that is controllable with respect to L and $A_{uc} \cup A_{uo}$.*

Proof. The same as the proof of Theorem 9.2 [20] with the only difference that A_{uc} is replaced now by $A_{uc} \cup A_{uo}$. □

The effect of the procedure described in Algorithm 1 is to remove the states of automaton S' that violate condition (iii). It is clear that Algorithm 1 stops after a finite number of iterations for regular languages K and L represented by finite automata S' and S, respectively. The representation $\hat{R}$ of E (supremal sublanguage of K that is controllable with respect to L and $A_{uc} \cup A_{uo}$) given by Algorithm 1 is often not suitable for the forthcoming algorithm. Notice that in Algorithm 1 it was not necessary that S' is state-partition automaton. However, Algorithm 1 must be followed by another algorithm in order to ensure that condition (ii) of normal relation holds and this property of representation will be required.

Let us suppose that S_1 is a representation of E such that S_1 is a subautomaton of S and S_1 is a state-partition automaton. This will be needed for the correctness of the algorithm below. Denote by $l_1 : S_1 \rightarrow \mathcal{L}$ the behavior homomorphism of S_1, i.e. $l_1(s_0)^1 = E$.

Remark 6.3. *Note that S can now be a different representation than the one resulted from Algorithm 1, because of the requirements that S_1 is a subautomaton of S and that S_1 is a state-partition automaton. Such representations S_1 as subautomaton of S can be constructed using the procedure*

from [7] that ensures the condition of S_1 being a state-partition automaton. It is not difficult to see that in automaton $S_1 = (S_1, \langle o_1, t_1\rangle)$ we have

$$(C)\ \forall s \in S_1 : \forall u \in A_{uc} \cup A_{uo} :\ (s \overset{u}{\rightarrow} \Rightarrow\ s \overset{u}{\rightarrow}_1).$$

This means that the controllability condition does not depend on the particular representation. This is very important, because in Algorithm 1 smaller representations of K and L can be used, while the condition (iii) of normal relations remains valid for representations S_1 and S that we use in Algorithm 2.

Thus we consider separately (ii) of normal relations in Algorithm 2 below. We denote by $Acc(S)$ the accessible part of an automaton S and make the following construction.

Algorithm 2. *Construct the automaton $(\tilde{S}, \langle\tilde{o}, \tilde{t}\rangle)$ in the following way.*
1) We put $\tilde{S} = S_1$.
2) $\tilde{t} : \tilde{S} \to (1 + \tilde{S})^A$ with
for $a \in A_{uo}$: $q \in \tilde{S}$: $\tilde{t}(q)(a)$ is defined iff $t_1(q)(a)$ is defined, in which case $\tilde{t}(q)(a) = t_1(q)(a)$, and:

$$\textit{for } a \in A_o :\ \ \tilde{t}(q)(a) \textit{ is defined iff } \forall s \in \lfloor q\rfloor_{Aux(S_1)} :\ (s \overset{a}{\rightarrow} \Rightarrow\ s \overset{a}{\rightarrow}_1),$$

in which case $\tilde{t}(q)(a) = t_1(q)(a)$.
3) The output function is unchanged: $\tilde{o} = o_1$.
4) Put $(\tilde{S}, \langle\tilde{o}, \tilde{t}\rangle) := Acc(\tilde{S}, \langle\tilde{o}, \tilde{t}\rangle)$.

Algorithm 2 just states that for any state $q \in \tilde{S}$ an outgoing edge labeled by $a \in A_o$ is to be removed (of course only if $q \overset{a}{\rightarrow}_1$), whenever there exists $s \in \lfloor q\rfloor_{Aux(S_1)} :\ \ s \overset{a}{\rightarrow}$ and $s \overset{a}{\not\rightarrow}_1$ in S_1. Intuitively it means that we remove an observable a-transition from all $s \in \lfloor q\rfloor_{Aux(S_1)}$ at the same time, which makes the resulting language normal. However, there might be a conflict if there exists $q' \in \tilde{S}$ such that $\langle q, q'\rangle \in Aux(S_1)$ (i.e. $q \in \lfloor q'\rfloor_{Aux(S_1)}$) and $\lfloor q\rfloor_{Aux(S_1)} \neq \lfloor q'\rfloor_{Aux(S_1)}$. Then it might happen that for some $s \in \lfloor q\rfloor_{Aux(S_1)} \cap \lfloor q'\rfloor_{Aux(S_1)}$ we should remove $a \in A_o$ from $t_1(q)$ (regarding from the class $\lfloor q\rfloor_{Aux(S_1)}$) and on the other hand we should keep it regarding from $\lfloor q'\rfloor_{Aux(S_1)}$. From this characterization the importance of the assumption that $Aux(S_1)$ is an equivalence relation is easily seen. Indeed, if $Aux(S_1)$ is an equivalence relation then for any $q' \in \tilde{S}$ such that $\langle q, q'\rangle \in Aux(S_1)$ we obtain according to Observation 4.4 that $\lfloor q\rfloor_{Aux(S_1)} = \lfloor q'\rfloor_{Aux(S_1)}$, i.e. the above conflict situation cannot happen. Nevertheless, a stronger condition for S_1 being a state-partition automaton is required to quarantee the supremality of the normal sublanguage represented by the resulting automaton $Acc(\tilde{S}, \langle\tilde{o}, \tilde{t}\rangle)$. Note also that the procedure described in Algorithm 2 can lead to removal of certain states that become inaccessible from the initial state after removing some observable transitions.

Algorithm 1 followed by Algorithm 2 ensure the conditions of normal relations are fulfilled. Unlike Algorithm 1, which consists in iterative application of the operator Γ, Algorithm 2 is a monolithic one. However there is an intrinsic difficulty concerning the computation of supremal controllable and normal sublanguages. It is possible that the diagonal relation of the resulting automaton $\tilde{S}$ might not be a control relation, i.e. condition (ii) of control relation might be violated due to the removal of some uncontrollable edges in Algorithm 2. If the supremal normal sublanguage is of interest (i.e. the condition in the definition of Γ is required to be valid only for $u \in A_{uo}$ instead of $u \in A_{uo} \cup A_{uc}$), Algorithm 2 does not affect what has been done in Algorithm

1 due to the fact that in our definition of normal relations we have separated the conditions for observable events (ii) and unobservable events (iii). Therefore in Algorithm 2 we remove only observable transitions from states in $\tilde{S}$ and condition (iii) will hold for $\tilde{S}$, i.e. its diagonal relation will be a normal relation on $\tilde{S} \times S$. However, if the supremal normal and controllable sublanguage is of interest, then Algorithm 2 may affect the condition (iii) for some $u \in A_o \cap A_{uc}$. Only in the case $A_o \subseteq A_c$, the supremal normal and controllable sublanguage is obtained after Algorithm 1 and Algorithm 2 have been applied. This explains why in general an iterative scheme that consists in consecutive repeating of these algorithms in the same way as in [6] or [5] is used. In the next section we show that it is not necessary to consider such an iterative scheme if we implement Algorithm 1 as a monolithic algorithm instead of an iteration. In the proof of the following theorem we use the notation $\rightarrow_\prime$ for transition function $\tilde{t}$ of $\tilde{S}$. Recall that transition functions t and t_1 of S and S_1 are denoted through $\rightarrow$ and $\rightarrow_1$, respectively.

Now we are interested in computation of supremal $(L,P)-$normal sublanguage of K. We need the following modification of Algorithm 1, where the condition in the definition of operator Γ is required only for all $u \in A_{uo}$ instead of for all $u \in A_{uc} \cup A_{uo}$.

Algorithm 1'. *Define the following operator that maps any diagonal relation $H \subseteq S' \times S$ to*

$$\Gamma(H) = \{\langle s, s\rangle \in H \mid \forall u \in A_{uo} : \ s \stackrel{u}{\rightarrow} \Rightarrow \ s \stackrel{u}{\rightarrow}_\prime \ \textit{and} \ \langle s_u, s_u\rangle \in H).\}$$

We put $\hat{R} = \cap_{i\geq 0}\, \Gamma^i(R)$.

Theorem 6.4. *Algorithm 1' followed by Algorithm 2 yields the supremal $(L,P)-$normal sublanguage of K in the following sense: $\tilde{l}(s_0)^1$, where $\tilde{l} : \tilde{S} \rightarrow \mathcal{L}$ is the unique behavior homomorphism, is the supremal (L,P)-normal sublanguage of K.*

Proof. The coinductive proof principle is used. Note that step 4) of Algorithm 2 is not used, because in fact the behavior homomorphism $\tilde{l}$ takes automatically care of the accessibility operation. First we show that $\tilde{l}(s_0)^1$ is a (L,P)-normal sublanguage of K. To prove the normality of $\tilde{l}(s_0)^1$ we show that the following relation is a normality relation on $\tilde{S} \times S$. Then $\tilde{l}(s_0)$ is (L,P)-normal sublanguage of K according to theorem 5.7. Since normality is a property of the prefix closure, this means that $\tilde{l}(s_0)^1$ is $(L,P)-$normal sublanguage of K in the classical framework.

$$R = \{\langle (s_0)_u, (s_0)_u\rangle \mid u \in \tilde{l}(s_0)^2\ \}.$$

Take a pair $\langle (s_0)_v, (s_0)_v\rangle \in R$ for some $v \in \tilde{l}(s_0)^2$.

(i) If $(s_0)_v \stackrel{a}{\rightarrow}_\prime$ for $a \in A$, then clearly by construction of Algorithm 2 $(s_0)_v \stackrel{a}{\rightarrow}$. It is clear from the definition of R that $\langle (s_0)_{va}, (s_0)_{va}\rangle \in R$.

(ii) Let $a \in A_o$ be such that $(s_0)_v \stackrel{a}{\rightarrow}$ and let there exist $s' \in \tilde{S}$: $s' \approx_{Aux(\tilde{S})} (s_0)_v$ with $s' \stackrel{a}{\rightarrow}_\prime$. By Lemma 4.1 there exist two strings $w, w' \in A^*$ such that $P(w) = P(w')$, $(s_0)_v = (s_0)_w$, and $s' = (s_0)_{w'} \stackrel{a}{\rightarrow}$. According to the construction of Algorithm 2 for any $s \approx_{Aux(S_1)} (s_0)_{w'}$ there must be $s \stackrel{a}{\rightarrow} \Rightarrow \ \ s \stackrel{a}{\rightarrow}_1$. In order to show that $(s_0)_v \stackrel{a}{\rightarrow}_\prime$ it must be that for any $q \approx_{Aux(S_1)} (s_0)_v$ there must be $q \stackrel{a}{\rightarrow} \Rightarrow \ \ q \stackrel{a}{\rightarrow}_1$. But using the fact that $Aux(S_1)$ is transitive (follows from 4.3) and the fact that $s' \approx_{Aux(\tilde{S})} (s_0)_v$ implies that $s' \approx_{Aux(S_1)} (s_0)_v$ we obtain that $\langle s', q\rangle \in Aux(S_1)$. But this just means that for any $q \approx_{Aux(S_1)} (s_0)_v$ we have $q \stackrel{a}{\rightarrow} \Rightarrow \ q \stackrel{a}{\rightarrow}_1$, i.e. $(s_0)_v \stackrel{a}{\rightarrow}_\prime$.

(iii) Let $a \in A_{uo}$ be such that $(s_0)_v \stackrel{a}{\rightarrow}$. Then according to Algorithm 1' we have $(s_0)_v \stackrel{a}{\rightarrow}_\prime$, because $a \in A_{uo}$ and $\tilde{S} \subseteq S_1$. This shows together with (i) and (ii) that R is a normality relation on $\tilde{S} \times S$.

We show finally that $\tilde{l}(s_0)^1$ is the supremal $(L,P)-$normal sublanguage of K. Let N be a $(L,P)-$normal sublanguage of K. We construct an auxiliary partial language $(N,\bar{N})$ that is for the sake of simplicity also denoted by N. Similarly partial languages corresponding to K and L, i.e. $(K,\bar{K})$ and $(L,\bar{L})$, respectively, are denoted by K and L. This abuse of notation should not lead to any confusion. Then it is sufficient to show that

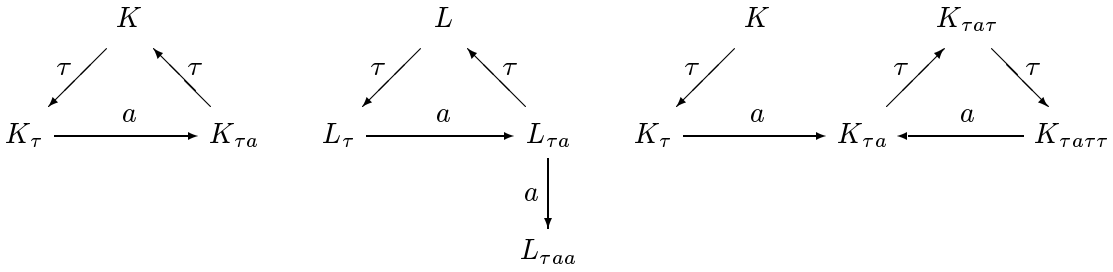
$$R = \{\langle N_u, \tilde{l}(s_0)_u\rangle \mid u \in N^2\}$$

is a simulation relation. Then we will have $(N,\bar{N}) \subseteq \tilde{l}(s_0)$, i.e. in particular $N \subseteq \tilde{l}(s_0)^1$. Take an arbitrary pair $\langle N_w, \tilde{l}(s_0)_w\rangle \in R$ for some $w \in N^2$.
(i) Let $N_w \downarrow$, i.e. $K_w \downarrow$, and therefore $\tilde{o}((s_0)_w) = o_1((s_0)_w) = 1$ according to point 3) of Algorithm 2. But this is equivalent to $\tilde{l}(s_0)_w \downarrow$ as a partial language.
(ii) Let $N_w \stackrel{a}{\rightarrow}$ for $a \in A_o$. Then also $K_w \stackrel{a}{\rightarrow}$, because $N \subseteq K$ (the inclusion holds for both ordinary and induced partial languages). Thus, $L_w \stackrel{a}{\rightarrow}$ as well. This means that $(s_0)_w \stackrel{a}{\rightarrow}_1$ and $(s_0)_w \stackrel{a}{\rightarrow}$. In order to show that $\tilde{l}(s_0)_w \stackrel{a}{\rightarrow}$ for $a \in A_o$, i.e. $(s_0)_w \stackrel{a}{\rightarrow}'$ we must prove that for any $q \approx_{Aux(S_1)} (s_0)_w$: $q \stackrel{a}{\rightarrow} \Rightarrow q \stackrel{a}{\rightarrow}_1$. There exist $v, v' :\ P(v) = P(v')$ such that $q = (s_0)_{v'}$ and $(s_0)_w = (s_0)_v$. Since S_1 is a state-partition automaton and $(s_0)_w$ is in two possibly different states of the observer automaton, we conclude by the property of state-partition automaton that these two states of the observer automaton coincide. But this means that there exists $w' \in A^*$ such that $P(w) = P(w')$ and $q = (s_0)_{w'}$. Now $q \stackrel{a}{\rightarrow}$ means that $w'a \in L^2$. Using normality of N it follows from $wa \in N^2$ and $w'a \in L^2$ that $w'a \in N^2$. Therefore $w'a \in K^2$ (because $N \subseteq K$), which means that $q \stackrel{a}{\rightarrow}_1$. The case $a \in A_{uo}$ is much easier. Again, $N_w \stackrel{a}{\rightarrow}$ implies that $K_w \stackrel{a}{\rightarrow}$. We show that $\tilde{l}(s_0)_w \stackrel{a}{\rightarrow}$, i.e. $(s_0)_w \stackrel{a}{\rightarrow}'$. It follows from Algorithm 1' that $(s_0)_w \stackrel{a}{\rightarrow}$ implies that $(s_0)_w \stackrel{a}{\rightarrow}_1$, whence $(s_0)_w \stackrel{a}{\rightarrow}'$, because Algorithm 2 does not affect transitions labeled by $a \in A_{uo}$.

Note that since N was an arbitrary $(L,P)-$normal sublanguage of K and $\tilde{l}(s_0)^1$ was shown to be a $(L,P)-$normal sublanguage of K, $\tilde{l}(s_0)^1$ must be the supremal $(L,P)-$normal sublanguage of K. □

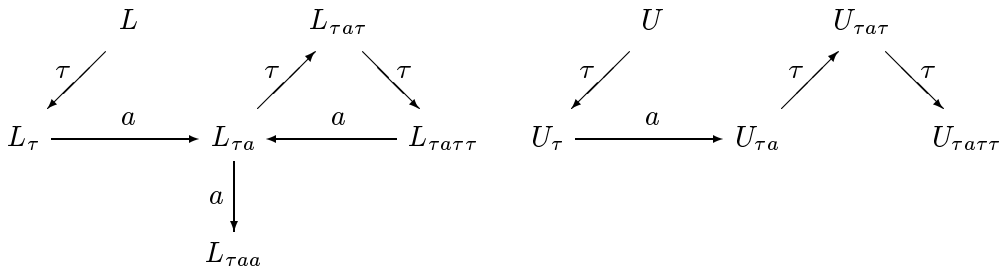
Now we illustrate the computation of the supremal $(L,P)-$normal sublanguage of K by the following simple example.

Example 3. *Let $A = \{a,\tau\}$ with $A_o = \{a\}$, K, and L are given by automata representations below. We assume that all states are marked, which does not play any role for normality.*



The original representation of K is not a state-partition automaton. The corresponding state-partition automaton representing the same language K, i.e. the synchronous product $S_1 := \langle K\rangle \parallel_{A_o} Obs(\langle K\rangle)$ is drawn on the right of K and L. The corresponding representation S of L such that S_1 is a subautomaton of S is given below together with the output of Algorithm 2,

denoted by U.



Notice that K is controllable with respect to L and A_{uo}, i.e. the action of Algorithm 1' is empty in this case. The Algorithm 2 then removes the second transition labelled by a, yielding thus U, which is easily seen to be the supremal $(L,P)-$normal sublanguage of K .

To conclude, we present a coalgebraic interpretation of the algorithm given in [7]. The algorithm given therein when interpreted in our framework yields a partial automaton $\check{S} = (\check{S}, \langle \check{o}, \check{t} \rangle)$, where $\check{S} = \{\cup_{d \in G} \, (s_0)_{\hat{d}}\}$ with $G = \{d \in P(K) : \ \forall s \in (s_0)_{\hat{d}} : \ \forall u \in A_{uo} : \ s \stackrel{u}{\to} \Rightarrow \ s \stackrel{u}{\to}_1\}$, which is the first part of the algorithm, and the definition of $\check{o} : \check{S} \to 2$ and $\check{t} : \check{S} \to (1+\check{S})^A$ is the second part. The output function is unchanged, i.e. $\check{o} = o_1|_{\check{S}}$. For any $q \in \check{S}$ there exists by definition of $\check{S}$ a $d \in G$: $s \in (s_0)_{\hat{d}}$. Using this,
for $q \in \check{S}$ and $a \in A_{uo}$ $\check{t}(q)(a)$ is defined iff $t_1(q)(a)$ is defined, in which case $\check{t}(q)(a) = t_1(q)(a)$, and

$$\text{for } a \in A_o : \quad \check{t}(q)(a) \text{ is defined iff } \forall s \in (s_0)_{\hat{d}} :$$

$$(s \stackrel{a}{\to} \Rightarrow \ \ s \stackrel{a}{\to}_1), \text{in which case } \check{t}(q)(a) = t_1(q)(a).$$

The construction of $\check{t}$ is the second part of their algorithm, where the condition for $\check{t}$ to be defined is similar to our condition 2 of Algorithm 2, but expressed using deterministic weak derivatives. An important feature is that S_1 is a state-partition automaton, thus $d \in P(K)^2$ such that $q \in (s_0)_{\hat{d}}$ is for any state in $q \in \tilde{S}$ uniquely determined. Remark that the last procedure which is taken from [7] is not correct if S_1 is not a state-partition automaton.

Our procedure for computation of the supremal $(L,P)-$normal sublanguage of K consists in Algorithm 1' followed by Algorithm 2. We show that it is different from that described in [7] in both steps. Implicitly, the two steps are also present in that paper, the first one is the construction of carrier set (removal of some states) and the second one consists in removing some observable transitions.

It has been shown that for regular languages K and L there exists always a finite automaton representation that satisfies the condition of state-partition automaton. Namely, the synchronized product $\langle K \rangle_P := \langle K \rangle \parallel Obs(\langle K \rangle)$ is a state-partition automaton as follows from results in [6]. In particular, $Aux(\langle K \rangle_P)$ is an equivalence relation according to Proposition 4.3.

Remark that this procedure is correct only if there is no conflict between different states of $Obs(S_1)$. This means that S_1 must be a state-partition automaton.

Finally, let us compare our algorithms with those from [7]. Our algorithm for computation of supremal normal sublanguages (Algorithm 1' followed by Algorithm 2) differs from that presented in [7] in both steps. The first step (construction of $\check{S}$) differs from our Algorithm 1' in that we do not necessarily remove in Algorithm 1' all states in $(s_0)_{\hat{d}}$ for $d \notin G$ as can be shown in a simple example. However these states are automatically removed by Algorithm 2, while producing the accessible part of $\tilde{S}$. This means that there is some saving on computational complexity using our

algorithm comparing to that presented in [7], because Algorithm 1 can use minimal representations, the computation of the set G and the automaton $\check{S}$ from [7] is not necessary. Nevertheless, both algorithms suffer from the exponential complexity in the worst case, because they rely on the subset constructions related to partial observations.

The concepts developed in this paper lead to new algorithms for supervisory control with partial observations presented without details in [8]. In that paper the concept of normality of a language with respect to a plant is also captured by different relations. These are introduced on finite automata representations in order to make the computations feasible. Our approach is inspired by the work of Cho and Marcus [7], where algebraic characterizations using the concept of invariant relations have been presented. The main advantage of the coalgebraic approach is that the formulations using relations provide a canonical way how to check different properties of languages (like controllability, observability, and normality). Since all these relations are in fact different weaker forms of bisimulation, we can proceed in the same way as for checking the bisimilarity [20]. Coalgebraic methods yield new algorithms and more general results for the computation of the supremal normal and normal and controllable (see also next section) sublanguages based on the corresponding relations.

Remark finally that the algorithm we have presented is composed of two separate algorithms, which makes its use in some theoretical problems involving supremal normal sublanguages (e.g. conditions for its commutation with synchronous product) quite difficult. Therefore we will present in the next section monolithic algorithms for the computation of the supremal normal and normal and controllable sublanguages motivated by coinductive definitions.

6.2 Note about maximal observable sublanguages

A procedure for computation of maximal observable sublanguages has been proposed by Cho and Marcus in [7]. It turns out that there are many technical difficulties while computing such maximals. The main issue is to ensure the procedure to be nonretrospective, i.e. that the procedure does not affect what has been computed earlier in the algorithm.

It is to be be expected that in our setting similar problems occur. An algorithm for maximal observable sublanguages can be designed using observability relations. However it is not easy to ensure the correctness of such a procedure because of the difficulties related to the fact that now some unobservable transitions are also to be removed. Moreover, it is not necessary to remove transitions simultaneously from all states that form a state of the observer automaton and there is no unique way how to do it, i.e. different orderings of controllable event set must be considered. In the next section another approach to the synthesis of observable sublanguages is presented. It will be based on coinductive definitions.

7 Coinduction in discrete-event control

This section is devoted to the application of a powerful technique called coinduction to discrete-event control. While coinductive proofs have already been used in the previous sections, coinductive definitions are used below to capture some important concepts like closed-loop language and optimal super/sublanguages.

7.1 Coinductive definition of supervised product and partial bisimulation under partial observations

In this subsection we present the definition of a supervised product of languages that describes the behavior of a supervised DES under partial observations. Assume throughout this section that the specification K and the open-loop partial language L ($K \subseteq L$) are given.

In the last section we have been studying relations on automata representations and we have formulated the basic properties of observability and normality using these relations. Now we aim at using the coinductive definitions. For this reason we must work with the final automaton of partial languages, where the coinductive definitions can be used. Coinductive definitions are used for defining algebraic operations, e.g. binary operations, on elements of final coalgebras. They consist in defining the coalgebraic structure (given by the functor) on the result of operation. For partial languages this means that new operations can be introduced (or sometimes the known ones reintroduced) by defining the output and transition functions (i.e. input derivatives). Interestingly, differential equations from analysis may also be viewed as coinductive definitions of solutions they define if a suitable coalgebraic structure (stream automata for ODEs, weighted automata for PDEs) is used [22].

Note that observability and normality relations can be defined in the final automaton $\mathcal{L}$. However, there is a difficulty with the fact that once we use the minimal representations $\langle K\rangle, \langle L\rangle \in \mathcal{L}$ as the subautomata of $\mathcal{L}$ generated by K and L, respectively, it is not true in general that for $K \subseteq L$, $\langle K\rangle$ is a subautomaton of $\langle L\rangle$. Therefore some additional technicalities are involved. In particular, $Aux(S_1)$ is replaced by $Aux(K, L)$ to stress the fact that both $\langle K\rangle$ and $\langle L\rangle$ are involved. Its definition has been first presented in [9].

In order to characterize the observability property we first need to introduce the following auxiliary relation defined on $DK \times DL$. Note that any relation $R \subseteq (DK \times DL)^2$ can be endowed with the following transition structure: for $a \in A$ $(M, N) \xrightarrow{a} (M', N')$ iff $M \xrightarrow{a} M_a$ and $N \xrightarrow{a} N_a$ with $M' = M_a$ and $N' = N_a$. We write $(M, N) \overset{P(a)}{\Rightarrow} (M', N')$ iff $\exists s \in M^2 \cap N^2$: $P(s) = a$, $M' = M_s$, and $N' = N_s$.

Definition 7.1. *A binary relation $Aux(K, L) \subseteq (DK \times DL)^2$, called* observational indistinguishability relation, *is the smallest relation satisfying:*

(i) $\langle (K, L), (K, L)\rangle \in Aux(K, L)$

(ii) *If* $\langle (M, N), (Q, R)\rangle \in Aux(K, L)$ *then* $\forall a \in A$: *if* $(M, N) \overset{P(a)}{\Rightarrow} (M', N')$ *and* $(Q, R) \overset{P(a)}{\Rightarrow} (Q', R')$ $\Rightarrow$ $\langle (M', N'), (Q', R')\rangle \in Aux(K, L)$

For $(M, N), (Q, R) \in DK \times DL$ we write $(M, N) \approx^{K,L}_{Aux} (Q, R)$ whenever $\langle (M, N), (Q, R)\rangle \in Aux(K, L)$.

Lemma 7.1. *For given partial languages K, L: $\langle (M, N), (Q, R)\rangle \in Aux(K, L)$ iff there exist two strings $s, s' \in K^2$ such that $P(s) = P(s')$ and $M = K_s$, $N = L_s$, $Q = K_{s'}$, and $R = L_{s'}$.*

Proof. ($\Leftarrow$) Let $(M, N) \in DK \times DL$ and $(Q, R) \in DK \times DL$ and there exist two strings $s, s' \in K^2$ such that $P(s) = P(s')$, $M = K_s$, $N = L_s$, $Q = K_{s'}$, and $R = L_{s'}$. Let $s = s_1 \dots s_n$ and $s' = t_1 \dots t_m$. Let $P(s) = P(s') = a_1 \dots a_k$. Then $n \geq k$, $m \geq k$, and there exist two increasing sequences of integers (indices) $u_i \geq i,\ i = 1, \dots, k$ and $v_i \geq i,\ i = 1, \dots, k$ such that $a_i = s_{u_i} = t_{v_i}$. Since $s, s' \in K^2$, and all a_i are observable events we can write $(K, L) \overset{P(a_1)\dots P(a_n)}{\Longrightarrow} (M, N)$ and also $(K, L) \overset{P(a_1)\dots P(a_n)}{\Longrightarrow} (Q, R)$, whence by (ii) inductively

applied $(M,N) \approx_{Aux}^{K,L} (Q,R)$.
$(\Rightarrow)$ Let $(M,N) \approx_{Aux}^{K,L} (Q,R)$. By the construction of $Aux(K,L)$ there exist $a_1,\ldots,a_k \in A$ such that $(K,L) \stackrel{P(a_1)\ldots P(a_k)}{\Longrightarrow} (M,N)$ and $(K,L) \stackrel{P(a_1)\ldots P(a_k)}{\Longrightarrow} (Q,R)$. Therefore there exist two strings s, s' with the same projection with $M = K_s$, $N = L_s$, $Q = K_{s'}$, and $R = L_{s'}$. □

Now we repeat the definition of the observability relation used in [9].

Definition 7.2. *(Observability relation.) Given two (partial) languages K and L, a binary relation $O(K,L) \subseteq DK \times DL$ is called an* observability relation *if for any $\langle M,N\rangle \in O(K,L)$ the following items hold:*

(i) $\forall a \in A:\ M \stackrel{a}{\rightarrow} \ \Rightarrow \ \ N \stackrel{a}{\rightarrow}$ *and* $\langle M_a, N_a\rangle \in O(K,L)$

(ii) $\forall a \in A_c:\ N \stackrel{a}{\rightarrow}$ *and* $(\exists M' \in DK, N' \in DL : (M',N') \approx_{Aux}^{K,L} (M,N)$ *and* $M' \stackrel{a}{\rightarrow}) \ \Rightarrow$ $M \stackrel{a}{\rightarrow}$ *and* $\langle M_a, N_a\rangle \in O(K,L)$.

For $M \in DK$ and $N \in DL$ we write $M \approx_{O(K,L)} N$ whenever there exists an observability relation $O(K,L)$ on $DK \times DL$ such that $\langle M,N\rangle \in O(K,L)$. In order to check whether for a given pair of (partial) languages (K and L), K is observable with respect to L, it is sufficient to establish an observability relation $O(K,L)$ on $DK \times DL$ such that $\langle K,L\rangle \in O(K,L)$. Indeed, we have

Theorem 7.2. *A (partial) language K is observable with respect to L (with $K \subseteq L$) and P iff $K \approx_{O(K,L)} L$.*

Proof. $(\Rightarrow)$ Let K be observable with respect to L. Denote

$$O_1(K,L) = \{\langle K_u, L_u\rangle \in DK \times DL \mid u \in K^2\ \}.$$

Let us show that $O_1(K,L)$ is an observability relation.
Let $\langle M,N\rangle \in O_1(K,L)$. We can assume that $M = K_s$ and $N = L_s$ for $s \in K^2$. We must show that conditions (i) and (ii) of the Definition 7.2 are satisfied.
(i) Let $M \stackrel{a}{\rightarrow}$ for $a \in A$. Notice that $K \subseteq L$ implies that for any $u \in K^2$, $K_u \subseteq L_u$. In particular $N \stackrel{a}{\rightarrow}$, because $M = K_s \subseteq L_s = N$ and it follows from the definition of $O_1(K,L)$ that $\langle M_a, N_a\rangle \in O_1(K,L)$.
(ii) Let $N \stackrel{a}{\rightarrow}$ for $a \in A_c$ and $\exists (M',N') \approx_{Aux}^{K,L} (M,N) \ \ : \ \ M' \stackrel{a}{\rightarrow}$. Then by Lemma 7.1 there exist two strings $s', s'' \in K^2$ such that $P(s') = P(s'')$ and $M' = K_{s'}$, $N' = L_{s'}$, $M = K_{s''} (= K_s)$, and $N = L_{s''} (= L_s)$. Now $M' \stackrel{a}{\rightarrow}$ implies that $s'a \in K^2$. From $N \stackrel{a}{\rightarrow}$ and $N = L_{s''}$ follows $s''a \in L^2$. Now by application of the observability of K with respect to L and P we deduce $s''a \in K^2$, i.e. $a \in K^2_{s''} = M^2$. This means that $M \stackrel{a}{\rightarrow}$, which was to be proved. The rest follows from (i).
$(\Leftarrow)$ Let $K \approx_{O(K,L)} L$. Let us show that K is observable with respect to L and P. For this purpose, let $s \in K^2$ and $a \in A_c$ such that $s'a \in K^2$ and $sa \in L^2$ and $P(s) = P(s')$. Then $s \in K^2 \cap L^2$, i.e. $L \stackrel{s}{\rightarrow}$ and $K \stackrel{s}{\rightarrow}$, whence from (i) of definition 7.2 inductively applied $K_s \approx_{O(K,L)} L_s$. Since $K \subseteq L$ and $s'a \in K^2$, we have $s' \in L^2$, because K^2 is prefix-closed. According to Lemma 7.1 we have $(K_s, L_s) \approx_{Aux}^{K,L} (K_{s'}, L_{s'})$. Notice that $sa \in L^2$ means $L_s \stackrel{a}{\rightarrow}$, and similarly $s'a \in K^2$ means $K_{s'} \stackrel{a}{\rightarrow}$. By (ii) of the definition of observability relation we obtain that $K_s \stackrel{a}{\rightarrow}$, i.e. $sa \in K^2$. □

Remark 7.3. *In the sequel we need also another type of auxiliary relations $Aux(S)$ for the special case $S = \langle K \rangle$. We will write $Aux(K)$ instead of $Aux(\langle K \rangle)$. Notice that it is possible to extend the definition of $Aux(S)$ to $Aux(Pwr(S))$ with the only difference, that the propagation of this relation is realized by unions of nondeterministic transitions, in particular by deterministic weak transitions. In the case of the final automaton of partial languages similar construction of observational indistinguishability relation is to be realized on $\mathrm{Pwr}(\mathrm{suffix}(K))$. Now we prepare the coinductive definition of the supervised product. This definition will consider arguments from $\mathrm{Pwr}(\mathrm{suffix}(K))$ and $\mathrm{Pwr}(\mathrm{suffix}(L))$ rather than from DK and DL. In fact we will work with unions of the form $\cup_{i=1}^{k} K_{s_i} \in \mathrm{Pwr}(\mathrm{suffix}(K))$, where $P(s_1) = \cdots = P(s_k)$. In order to keep the notation simple, we will use an extension of $Aux(K)$ to such unions of derivatives. In the definition of supervised product this will be needed.*

Now we give a formal definition of $Aux(K)$ extended to $\mathrm{Pwr}(\mathrm{suffix}(K))$.

Definition 7.3. *(Extension of $Aux(K)$ from DK to Pwr(suffix(K))). A binary relation $Aux(K) \subseteq (\mathrm{Pwr}(\mathrm{suffix}(K)))^2$, called* observational indistinguishability relation *is the smallest relation satisfying:*

(i) $\langle (K, K) \in Aux(K)$

(ii) *If $\langle M, N \rangle \in Aux(K)$ then $\forall a \in A:\ M \stackrel{a}{\to} M_a$ and $N \stackrel{a}{\to} N_a \ \Rightarrow\ \langle M_a, N_a \rangle \in Aux(K)$*

(iii) *If $\langle M, N \rangle \in Aux(K)$ then $\forall m, n \in Z_+$: if $M \stackrel{\varepsilon}{\Rightarrow} M_1, M \stackrel{\varepsilon}{\Rightarrow} M_2, \ldots, M \stackrel{\varepsilon}{\Rightarrow} M_n$, and $N \stackrel{\varepsilon}{\Rightarrow} N_1, \ldots, N \stackrel{\varepsilon}{\Rightarrow} N_m$, then $\langle \cup_{i=1}^{n} M_i, \cup_{j=1}^{m} N_j \rangle \in Aux(K)$.*

Clearly, a natural extension of Lemma 4.1 holds. Namely, $\langle \cup_{i=1}^{k} K_{s_i}, \cup_{j=1}^{l} L_{t_j} \rangle \in Aux(K)$, where $P(s_1) = \cdots = P(s_k)$ and $P(t_1) = \cdots = P(t_l)$ iff $P(s_1) = P(t_1)$, which implies naturally $P(s_i) = P(t_j)\ \forall i, j$. The notation $\cup_{i=1}^{k} K_{s_i} \approx_{Aux}^{K} \cup_{j=1}^{l} L_{t_j}$ is also used.

Definition 7.4. *(Supervised product under partial observations.) Define the following binary operation on (partial) languages called supervised product under partial observations for all $M \in \mathrm{Pwr}(\mathrm{suffix}(K))$ and $N \in \mathrm{Pwr}(\mathrm{suffix}(L))$:*

$$(M /\!\!/_U^O N)_a =$$

(1) $M_a /\!\!/_U^O N_a$ *if* $M \stackrel{a}{\to}$ *and* $N \stackrel{a}{\to}$*;*

(2) $(\cup_{\{M' : \langle M', M \rangle \in Aux(K)\}} M'_a) /\!\!/_U^O N_a$ *if* $M \stackrel{a}{\not\to}$ *and* $\exists M' \in DK$: $M' \approx_{Aux}^{K} M$ *such that* $M' \stackrel{a}{\to}$ *and* $N \stackrel{a}{\to}$ *and* $a \in A_c \cup A_o$*;*

(3) $0 /\!\!/_U^O N_a$ *if* $M \stackrel{a}{\not\to}$ *and* $\forall M' \in DK\ : M' \approx_{Aux}^{K} M:\ M' \stackrel{a}{\not\to}$ *and* $N \stackrel{a}{\to}$ *and* $a \in A_{uc} \cap A_o$*;*

(4) $M /\!\!/_U^O N_a$ *if* $M \stackrel{a}{\not\to}$ *and* $N \stackrel{a}{\to}$ *and* $a \in A_{uc} \cap A_{uo}$*;*

(5) $\emptyset$ *otherwise*

and $(M /\!\!/_U^O N)\downarrow$ *iff* $N \downarrow$.

Remark 7.4. *1. According to Observation 2.4, $DL \subseteq \mathrm{Pwr}(\mathrm{suffix}(L))$ and since $K \subseteq L$ also $DK \subseteq \mathrm{Pwr}(\mathrm{suffix}(L))$.*
2. It follows from the definition of supervised product that $K \subseteq (K /\!\!/_U^O L) \subseteq L$. Both inclusions can be verified by construction of the corresponding simulation relations. Let us show that $K \subseteq (K /\!\!/_U^O L)$. Consider the following relation:

$$R(K,L) = \{\langle K_w, (K/^O_U L)_w\rangle \mid w \in K^2\}.$$

It easy to see that $R(K,L)$ is a simulation relation proving the claimed inclusion. Take $w \in K^2$.
(i) If $K_w \downarrow$, then $w \in K^1$, i.e. $w \in L^1$, which means $L_w \downarrow$. Furthermore, it follows from the definition of 7.4 that $(K/^O_U L)_w = K_w/^O_U L_w$. Therefore, $(K/^O_U L)_w \downarrow$.
(ii) if for $a \in A$: $K_w \xrightarrow{a}$, then $(K/^O_U L)_{wa} = (K_w/^O_U L_w)_a = K_{wa}/^O_U L_{wa}$, i.e. $(K/^O_U L)_w \xrightarrow{a}$ and $\langle K_w, (K/^O_U L)_w\rangle \in R(K,L)$.

As a consequence we conclude that the range of supervised product is again $\mathrm{Pwr}(\mathrm{suffix}(L))$, *i.e. the supervised product can be also viewed as a (partial) binary operation on* $\mathrm{Pwr}(\mathrm{suffix}(L))$.

The definition of supervised product under partial observations is quite complicated due to the interconnections between observability and controllability that must be taken into account. It deserves additional comments. Notice that several cases must be distinguished. First of all, by (1) the controller allows any event that does not exit from its (supervisor) language. A controllable event is enabled when the supervisor observes $s \in A^*$ iff there exists a string with the same projection as s that can be continued by this event within the supervisor's language, which is included in (2). The controller also enables all uncontrollable events that are possible in the plant, but the future actions depend on whether the occurred uncontrollable event is observable or not. If the uncontrollable event is unobservable then the first component of the supervised product need not to move, but only the second component is updated as is seen from (4) above. In the case that the uncontrollable event a is observable, there must be further specified whether there exists a derivative indistinguishable from a derivative currently considered that can make an $a-$transition (i.e. there exists a string that has the same projection as s that can continued by a within the supervisor's language), in which case the action is the same as for controllable events (i.e. this case is included in (2) above), or whether there is no such derivative, which means that only uncontrollable events that are possible in the plant are allowed in the future. The latter case corresponds to the term containing the zero partial language and is labeled by (3) above. In any other case (5) the controllable events are disabled by the supervisor. We have thus the coinductive definition of the closed-loop language that gives a clear picture of what is the mechanism of discrete-event control under partial observations.

Note that a similar attemp to capture the behavior of the interaction of a supervisor with a plant has been made in [12], where this interaction is represented by the so called masked prioritized synchronization. Although we can see a similar classification of event types (with respect to their controllability and observability) as in our supervised product, the setting of that paper is somewhat different: considers priority sets for both plant and the supervisor and an interface masks.

Now we proceed in the same way as in the case of full observations. Let us define the following relation called partial bisimulation under partial observations.

Definition 7.5. *(Partial bisimulation.) A binary relation $R(K,L) \subseteq DK \times DL$ is called a* partial bisimulation under partial observations *if for all $\langle M, N\rangle \in R(K,L)$:*

(i) $o(M) = o(N)$ *($M\downarrow$ iff $N\downarrow$)*

(ii) $\forall a \in A:\ M \xrightarrow{a} \Rightarrow N \xrightarrow{a}$ *and* $\langle M_a, N_a\rangle \in R(K,L)$

(iii) $\forall u \in A_{uc}:\ N \xrightarrow{u} \Rightarrow M \xrightarrow{u}$ *and* $\langle M_u, N_u\rangle \in R(K,L)$

(iv) $\forall a \in A_c:\ N \xrightarrow{a}$ *and* $(\exists (M', N') \approx^{K,L}_{Aux} (M,N):\ M' \xrightarrow{a}) \Rightarrow M \xrightarrow{a}$.

For $M \in DK$ and $N \in DL$ we write $M \approx_U^{O(K,L)} N$ whenever there exists a partial bisimulation under partial observations $R(K,L)$ such that $\langle M, N\rangle \in R(K,L)$. This relation is called partial bisimilarity under partial observations.

Remark 7.5. *Notice that (i) relates the marking components of the languages involved and (ii) corresponds to the language simulation (inclusion), while (iii) to the controllability and (iv) to the observability condition. Observe also that the second statement on the right hand side of (iii) follows from the corresponding first statement and (ii).*

Now we are ready to formulate the main theorem, which gives a coalgebraic formulation of the controllability and observability theorem [5] in supervisory control of DES with partial observations.

Theorem 7.6. *Let $K \subseteq L$ are given partial languages. Then $K \approx_U^{O(K,L)} L$ iff $K = K/_U^O L$. The supervised product under partial observations of the languages K and L equals K iff K and L are partially bisimilar in the sense of Definition 7.5.*

Proof. ($\Rightarrow$) Let $K \approx_U^{O(K,L)} L$. Define

$$R(K,L) = \{\langle M, (M/_U^O N)\rangle \mid M \in DK,\ N \in DL \text{ and } M \approx_U^{O(K,L)} N\}.$$

According to the coinduction proof principle it is sufficient to prove that $R(K,L)$ is a bisimulation, because then $K \approx_U^{O(K,L)} L$, i.e. $\langle K, (K/_U^O L)\rangle \in R(K,L)$ implies that $K = (K/_U^O L)$. Let $\langle M, (M/_U^O N)\rangle \in R(K,L)$.
(i) $M\downarrow$ iff $N\downarrow$ (because $M \approx_U^{O(K,L)} N$) iff $(M/_U^O N)\downarrow$.
(ii) If $M \stackrel{a}{\rightarrow}$ for $a \in A$ then by (ii) of definition 7.5 $N \stackrel{a}{\rightarrow}$ and $M_a \approx_U^{O(K,L)} N_a$. Thus, $(M/_U^O N) \stackrel{a}{\rightarrow} (M/_U^O N)_a = (M_a/_U^O N_a)$, and $\langle M_a, (M/_U^O N)_a\rangle \in R(K,L)$.
(iii) If $(M/_U^O N) \stackrel{a}{\rightarrow}$, then according to the (coinductive) definition of the supervised product we have four possibilities : either $M \stackrel{a}{\rightarrow}$ and $N \stackrel{a}{\rightarrow}$, or $M \stackrel{a}{\not\rightarrow}$ and $\exists M' \approx_{Aux}^K M : M' \stackrel{a}{\rightarrow}$ and $N \stackrel{a}{\rightarrow}$ and $a \in A_c \cup A_o$, or $M \stackrel{a}{\not\rightarrow}$ and $\forall M' \in DK : M' \approx_{Aux}^K M : M' \stackrel{a}{\not\rightarrow}$ and $N \stackrel{a}{\rightarrow}$ and $a \in A_{uc} \cap A_o$; or, finally, $M \stackrel{a}{\not\rightarrow}$ and $N \stackrel{a}{\rightarrow}$ and $a \in A_{uc} \cap A_{uo}$. Notice however that the second case is contradicted by (iv) of definition 7.5: it is sufficient to see that if $\exists M' \approx_{Aux}^K M : M' \stackrel{a}{\rightarrow}$ and $N \stackrel{a}{\rightarrow}$, then $\exists (M', N') \approx_{Aux}^{K,L} (M, N) : M' \stackrel{a}{\rightarrow}$ and $N \stackrel{a}{\rightarrow}$. Indeed, by Lemma 4.1 applied for $S = DK$ (recall that $M \in DK$) $M = K_s$ and $M' = K_{s'}$ for some $s, s' : P(s') = P(s)$, then it is sufficient to put $N' = L_{s'}$, which clearly exists, because $K \subseteq L$. The third and the fourth cases (with $a \in A_{uc}$) are both impossible due to (iii) of the same definition. Hence only the first possibility can occur, which brings us back to the previous case (ii).
($\Leftarrow$) Let us show that the following relation is a partial bisimulation under partial observations. Define

$$T(K,L) = \{\langle M, N\rangle \mid M \in DK,\ N \in DL \text{ and } M = (M/_U^O N)\}.$$

Let $\langle M, N\rangle \in T(K,L)$.
(i) $M\downarrow$ iff $(M/_U^O N)\downarrow$ (from the definition of $T(K,L)$) iff $N\downarrow$ (from definition 7.4).
(ii) If $M \stackrel{a}{\rightarrow}$ for $a \in A$ then $(M/_U^O N) \stackrel{a}{\rightarrow}$ and clearly (from the coinductive definition of supervised product) $N \stackrel{a}{\rightarrow}$. Also $M_a = (M/_U^O N)_a = (M_a/_U^O N_a)$, whence $\langle M_a, N_a\rangle \in T(K,L)$.
(iii) If $N \stackrel{u}{\rightarrow}$ for $u \in A_{uc}$ then $(M/_U^O N) \stackrel{u}{\rightarrow}$ according to the definition of supervised product. Thus $M \stackrel{u}{\rightarrow}$ as well. Furthermore, $M_u = (M/_U^O N)_u = (M_u/_U^O N_u)$, which means $\langle M_u, N_u\rangle \in T(K,L)$.

(iv) If $N \stackrel{a}{\rightarrow}$ for $a \in A_c$ and ($\exists (M', N') \approx^{K,L}_{Aux} (M, N) \ : \ M' \stackrel{a}{\rightarrow}$) then from the definition of supervised product (the second case occurs: note that in particular $M' \approx^{K}_{Aux} M$), $(M/^{O}_{U} N) \stackrel{a}{\rightarrow}$, i.e. $M \stackrel{a}{\rightarrow}$, which was to be shown. □

Finally, similarly as in the case of full observations, there is the following characterization of partial bisimilarity.

Corollary 7.7. *$K \approx^{O(K,L)}_{U} L$ iff $(K^2 A_{uc} \cap L^2 \subseteq K^2$, $K \approx_{O(K,L)} L$, and $K^1 = K^2 \cap L^1)$.*

Proof. It is quite analogous to the full observations case. In particular, notice that partial bisimulation under partial observations implies partial bisimulation as it has been first introduced in [20]. Thus, it is sufficient to consider only the additional property of observability, which appears in both sides of the claimed equivalence. □

7.2 Infimal closed observable superlanguages and maximal observable sublanguages

This subsection contains only new results. In the last subsection we have introduced an operation on partial languages called supervised product under partial observations. This operation corresponds to the behavior of the supervised discrete-event system modeled by a partial automaton using the centralized version of $C\&P$ control architecture in the terminology of [34]. We call this control architecture in the centralized case simply permissive. Let $K = (K^1, K^2)$ be the desired behavior (partial language) and V be the supervisory controller. Then $\forall s \in A_o^*$ the associated control law (events enabled after V observes s) is:

$$\gamma_P(V, s) = A_{uc} \cup \{a \in A_c : \ \exists s' \in K^2 \text{ with } P(s') = P(s) \text{ and } s'a \in K^2\}.$$

The centralized counterpart of the $D\&A$ control architecture we call antipermissive and it is given by the following control law: $\forall s \in A_o^*$

$$\gamma_A(V, s) = A_{uc} \cup \{a \in A_c : \ \forall s' \in K^2 \text{ with } P(s') = P(s) \text{ we have } s'a \in L^2 \ \Rightarrow \ s'a \in K^2\}.$$

There is also an antipermissive control architecture counterpart of the supervised product, but its definition is postponed towards the end of this subsection. Let us call it antipermissive supervised product. We will show that it cannot be defined by coinduction, however in the very similar way using suitable automata representations. Note that the permissiveness or antipermissiveness is related to the observability (controllable events). Recall that the control policy must be, by definition, permissive with respect to uncontrollable events in the sense that these are always enabled.

Remark 7.8. *We consider from now on an order relation on partial languages induced by their second component only, i.e. we write $K \subseteq L$ iff $K^2 \subseteq L^2$. The same applies for infimum (supremum), and maximum operations. Note that only the second condition of simulation relations must be checked to prove such defined inclusion of partial languages.*

Let us recall the coinductive definition of the supervised product in the case of full observations from [20].

Definition 7.6. *Define the following binary operation on (partial) languages for all $K, L \in \mathcal{L}$ and $\forall a \in A$:*

$$(K/_U L)_a = \begin{cases} K_a/_U L_a & \text{if } K \xrightarrow{a} \text{ and } L \xrightarrow{a} \\ 0/_U L_a & \text{if } K \not\xrightarrow{a} \text{ and } L \xrightarrow{a} \text{ and } a \in A_{uc} \\ \emptyset & \text{otherwise} \end{cases}$$

and $(K/_U L)\downarrow$ iff $L\downarrow$.

Theorem 7.9. *$(K/_U L) = \inf(\bar{C}(K, L)) = \inf\{M \supseteq K : M$ is controllable with respect to L and $A_{uc}\}$, i.e. $K/_U L$ equals the infimal controllable superlanguage of K.*

Proof. Let us show that $K/_U L$ is a superlanguage of K that is controllable with respect to L and A_{uc}. It is clear from the definition of supervised product that $K \subseteq (K/_U L)$ in the sense of Remark 7.8. Let us show that $K/_U L$ is controllable with respect to L and A_{uc}. It is sufficient to prove that the following relation is a control relation.

$$C = \{\langle (K/_U L), L\rangle \mid K, L \in \mathcal{L}\}.$$

(i) Let $(K/_U L) \xrightarrow{a}$ and $L \xrightarrow{a}$ for $a \in A$. Then by coinductive definition of $K/_U L$ either $(K/_U L)_a = (K_a/_U L_a)$ or $(K/_U L)_a = (0/_U L_a)$. However, by definition of C in both cases we have $\langle (K/_U L)_a, L_a\rangle \in C$.
(ii) If $L \xrightarrow{u}$ for $u \in A_{uc}$, then either $K \xrightarrow{u}$ and hence $(K/_U L) \xrightarrow{u}$ or $K \not\xrightarrow{u}$, but according to the definition of $K/_U L$ we have still $(K/_U L) \xrightarrow{u} (0/_U L_u)$.

It remains to show the infimality. Let $M \supseteq K$ be controllable with respect to L and A_{uc}.

$$R = \{\langle (K/_U L), M\rangle \mid K, L, M \in \mathcal{L} : K \subseteq M \subseteq L, \text{ and } M^2 A_{uc} \cap L^2 \subseteq M^2\}.$$

satisfies (ii) of the definition of simulation relations. Let $(K/_U L) \xrightarrow{a}$ for $a \in A$. According to the definition of $K/_U L$ we have two possibilities: either $K \xrightarrow{a}$ and $L \xrightarrow{a}$, in which case $(K/_U L)_a = K_a/_U L_a$ or $K \not\xrightarrow{a}$ and $L \xrightarrow{a}$ and $a \in A_{uc}$. In the first case we have $M \xrightarrow{a}$ simply because $K \xrightarrow{a}$ and $K \subseteq M$, while in the latter case we have $M \xrightarrow{a}$ because of the controllability of M with respect to L and A_{uc} (by definition 5.3 of control relations for $a \in A_{uc}$: $L \xrightarrow{a} \Rightarrow M \xrightarrow{a}$). Moreover in both cases $\langle (K/_U L)_a, L_a\rangle \in R$. □

Although the infimal controllable superlanguages are important [13], supremal controllable sublanguages are even more interesting as least restrictive solutions of full observation supervisory control problems [32]. In [20] an algorithm for the computation of supremal controllable sublanguages, based on control relations, has been presented. It turns out that it is also possible to define the supremal controllable sublanguage by coinduction.

Definition 7.7. *Define the following binary operation on (partial) languages for all $K, L \in \mathcal{L}$ and $\forall a \in A$:*

$$(K/^S_C L)_a = \begin{cases} K_a/^S_C L_a & \text{if } K \xrightarrow{a} \text{ and } L \xrightarrow{a} \\ & \text{and if } \forall u \in A^*_{uc} : L_a \xrightarrow{u} \Rightarrow K_a \xrightarrow{u} \\ \emptyset & \text{otherwise} \end{cases}$$

and $(K/^S_C L)\downarrow$ iff $L\downarrow$.

Theorem 7.10. *$(K/^S_C L) = \sup(\underline{C}(K, L)) = \sup\{M \subseteq K : M$ is controllable with respect to L and $A_{uc}\}$, i.e. $K/^S_C L$ equals the supremal controllable sublanguage of K.*

Proof. First we show that $K/^S_C L$ is a sublanguage of K that is controllable with respect to L and A_{uc}. It is clear from the definition of $K/^S_C L$ that $(K/^S_C L) \subseteq K$ in the sense of Remark 7.8. Indeed, if we take $U = (K/^S_C L)_w = K_w/^S_C L_w$ and $V = K_w$ for some $w \in (K/^S_C L)^2$, then $U \xrightarrow{a} \Rightarrow V \xrightarrow{a}$. Let us show that $K/^S_C L$ is controllable with respect to L and A_{uc}. It is sufficient to prove that the following relation is a control relation (Definition 5.3).

$$C = \{\langle (K/^S_C L)_w, L_w\rangle \mid w \in (K/^S_C L)^2\}.$$

Take a pair $M = (K/^S_C L)_s$ and $N = L_s$ for some $s \in (K/^S_C L)^2$.
(i) Let $(K/^S_C L)_s \xrightarrow{a}$ and $L_s \xrightarrow{a}$ for $a \in A$. Then by coinductive definition of $K/^S_C L$ we have $(K/^S_C L)_{sa} = (K_{sa}/^S_C L_{sa})$, which by definition of C means that $\langle (K/^S_C L)_{sa}, L_{sa}\rangle \in C$.
(ii) Let $L_s \xrightarrow{u}$ for $u \in A_{uc}$. Since $(K/^S_C L) \xrightarrow{s}$, we have by definition 7.7 that $K \xrightarrow{s}$ and $L \xrightarrow{s}$ and $\forall u \in A^*_{uc}: \; L_s \xrightarrow{u} \Rightarrow K_s \xrightarrow{u}$. Therefore we deduce $K_s \xrightarrow{u}$. Furthermore, $\forall v \in A^*_{uc}$: $L_{su} \xrightarrow{v} \Rightarrow L_s \xrightarrow{uv} \Rightarrow K_s \xrightarrow{uv} \Rightarrow K_{su} \xrightarrow{v}$, because $uv \in A^*_{uc}$ and $(K/^S_C L) \xrightarrow{s}$. Hence $(K/^S_C L)_s \xrightarrow{u}$, which proves that C is a control relation, i.e. $K/^S_C L$ is controllable with respect to L and A_{uc}.

It remains to show the supremality. Let $M \subseteq K$ be controllable with respect to L and A_{uc}. In order to show that $M^2 \subseteq (K/^S_C L)^2$, we consider

$$R = \{\langle M_w, (K/^S_C L)_w\rangle \mid w \in M^2\}.$$

Take $\langle M_s, (K/^S_C L)_s\rangle \in R$ for some $s \in M^2$. Let $M_s \xrightarrow{a}$ for $a \in A$. Then $K_s \xrightarrow{a}$, and $L_s \xrightarrow{a}$, since $M \subseteq K \subseteq L$. In order to prove that $(K/^S_C L)_s \xrightarrow{a}$, it remains to show that $\forall u \in A^*_{uc}: \; L_{sa} \xrightarrow{u} \Rightarrow K_{sa} \xrightarrow{u}$. But this is straightforward: if $L_{sa} \xrightarrow{u}$, then by controllability of M we deduce $M_{sa} \xrightarrow{u}$, thus from $M \subseteq K$ it follows that $K_{sa} \xrightarrow{u}$. It follows that R satisfies (ii) of simulation relations, i.e. $M \subseteq K/^S_C L$. □

Let us now suppose that controllability is not an issue. Recall that an algorithm for supremal controllable sublanguage has been given in [20]. We have also shown that the supervised product in the case of full observations defined therein provides the infimal controllable superlanguage. As a byproduct we have its coinductive definition. In the case of partial observations, we can now separate the issue of controllability from observability and introduce the following modification of supervised product. Note that a similar method (separating the issue of controllability from observability) has been used in [1] for automata (supervisor) approach. Unlike the methods known from the literature ([1] and [24]) for infimal closed and observable superlanguages our coalgebraic approach (the following coinductive definition) has a direct algorithmic character, because coinduction defines the resulting structure event by event).

Definition 7.8. *Define the following binary operation on (partial) languages for all* $M \in \mathrm{Pwr}(\mathrm{suffix}(K))$ *and* $N \in \mathrm{Pwr}(\mathrm{suffix}(L))$ *and* $\forall a \in A$*:*

$$(M/^O N)_a = \begin{cases} M_a/^O N_a & \text{if } M \xrightarrow{a} \text{ and } N \xrightarrow{a} \text{ and} \\ & \exists s \in K^2 : \; M = K_s \text{ and } N = L_s \\ \cup_{\{M' : \langle M', M\rangle \in Aux(K)\}} M'_a/^O N_a & \text{if } M \not\xrightarrow{a} \text{ and } \exists M' \in DK : \\ & M' \approx^K_{Aux} M \text{ such that } M' \xrightarrow{a} \\ & \text{and } N \xrightarrow{a} \text{ and } a \in A_c \\ \emptyset & \text{otherwise} \end{cases}$$

and $(M/^O N)\downarrow$ *iff* $N\downarrow$.

The new operation has the following pleasant property:

Theorem 7.11. $(K/^O L) = \inf(\bar{O}(K, L, P)) = \inf\{M \supseteq K : \ M$ *is observable with respect to* L *and* $P\}$. *The infimal observable superlanguage of* K *equals* $(K/^O L)$.

Proof. It can be proven by coinduction using the formula for $\inf(\bar{O}(K, L, P))$ given in [24]. Another, more direct, way is to show that $K/^O L$ is an observable partial language containing K that is smaller then any other observable superlanguage of K.

Let us show that $K/^O L$ is a superlanguage of K that is observable with respect to L. It is clear from the definition 7.8 that $(K/^O L)^2$ is a superlanguage of K^2. Formally it can be checked by constructing an obvious simulation relation. Let us show that $K/^O L$ is observable with respect to L. According to theorem 7.2 we put

$$O = \{\langle (K/^O L)_u, L_u \rangle \mid u \in (K/^O L)^2 \ \}$$

and show that O is an observability relation on $D(K/^O L) \times DL$. Take a pair $\langle U, V \rangle \in R$. We can assume that $U = (K/^O L)_s$ and $V = L_s$ for some $s \in (K/^O L)^2$.
(i) Let $a \in A$ such that $(K/^O L)_s \xrightarrow{a}$. It follows from the definition 7.8 that $L_s \xrightarrow{a}$ and from the definition of O that $\langle (K/^O L)_{sa}, L_{sa} \rangle \in O$.
(ii) Let $a \in A_c$ such that $L_s \xrightarrow{a}$ and there exists $M \in D(K/^O L) : M \approx_{Aux}^{K/^O L, L} (K/^O L)_s$ with $M \xrightarrow{a}$. It means that there exist $s', s'' \in A^*$ such that $P(s'') = P(s')$, $(K/^O L)_s = (K/^O L)_{s''}$, $L_s = L_{s''}$, and $M = (K/^O L)_{s'} \xrightarrow{a}$. According to definition 7.8 inductively applied there exist $s_i \in A^*, \ i \in I$ such that $(K/^O L)_{s'} = (\cup_{i \in I} K_{s_i})/^O L_{s'}$, where $P(s_i) = P(s') \ \forall i \in I$. Notice, that it can be that $I = \{1\}$ and $s_1 = s'$. Since $M \xrightarrow{a}$, by definition 7.8 either there exist $s_j, \ j \in J \subseteq I$ such that $K_{s_j} \xrightarrow{a}$ for $j \in J$ and $M_a = (\cup_{j \in J} K_{s_j a})/^O L_{s'a}$, or there exist $w_k, \ k \in K$ such that $K \xrightarrow{w_k a}$, $P(w_k) = P(s')$ and $M_a = (\cup_{k \in K} K_{w_k a})/^O L_{s'a}$. Since also $P(w_k) = P(s'')$ for all $k \in K$, we deduce finally that according to definition 7.8 in both cases there must be $(K/^O L)_s = (K/^O L)_{s''} \xrightarrow{a}$, which proves that O is an observability relation.

The last step of the proof is to show that if $M \supseteq K$ is a language which is observable with respect to L and P, then $(K/^O L) \subseteq M$. It is sufficient to prove that

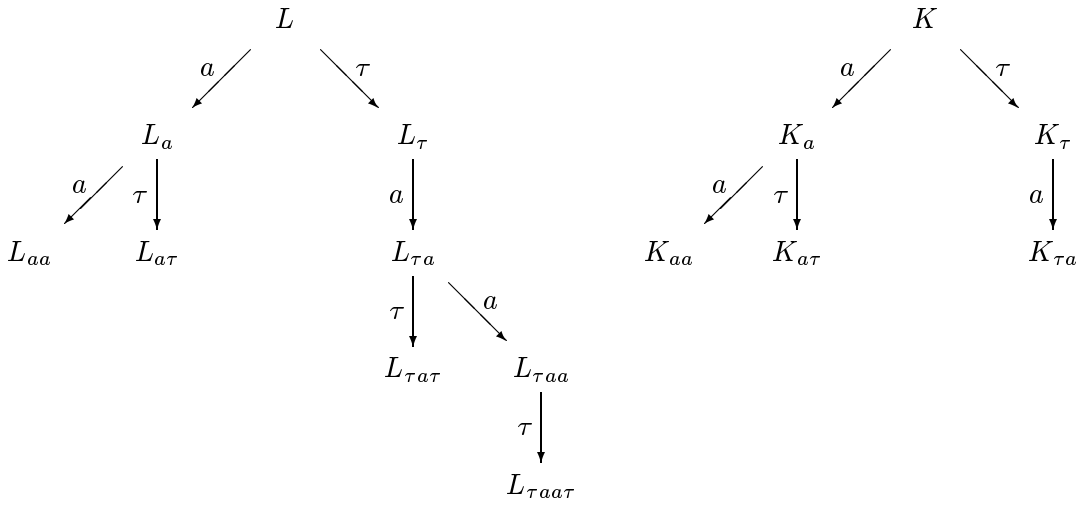
$$R = \{\langle (K/^O L)_u, M_u \rangle \mid u \in (K/^O L)^2 \text{ and } K \subseteq M \approx_{O(M,L)} L\}$$

satisfies (ii) of simulation relation.

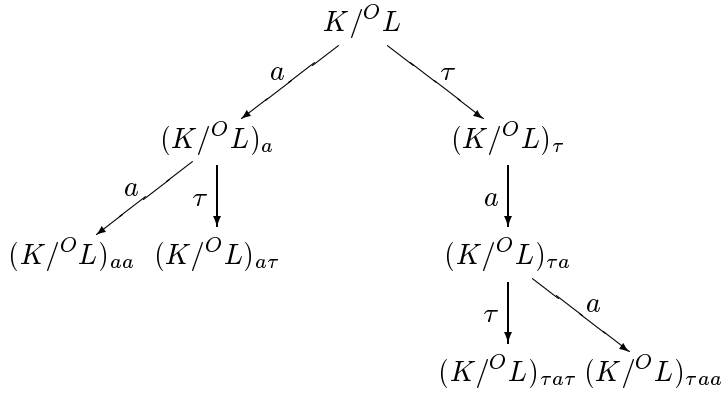
Take a pair $\langle U, V \rangle \in R$. We can assume that $U = (K/^O L)_w$ and $V = M_w$ for some $w \in (K/^O L)^2$. Let $U \xrightarrow{a}$. There exist $s_i \in K^2$ for i in some index set I such that $P(s_i) = P(w) \ \forall i \in I$ and $U = (\cup K_{s_i})/^O L_w$. Now, $U \xrightarrow{a}$ implies that either $U = K_w/^O L_w \xrightarrow{a} K_{wa}/^O L_{wa}$ or there exists $J \subseteq I$ such that $K_{s_j} \xrightarrow{a}$ for $j \in J$ and $U_a = (\cup K_{s_j a})/^O L_{wa}$ and $a \in A_c$ or finally there exist $w_k \in A^*, \ k \in K$ such that $P(w_k) = P(w)$, $a \in A_c$, and $U_a = (\cup K_{w_k a})/^O L_{wa}$. In the first case we have directly $wa \in K^2$, i.e. $V = M_w \xrightarrow{a}$. In the second case $w \in M^2$ (because $V = M_w$), $s_j \in M^2$, because $s_j \in K^2 \subseteq M^2$, $s_j a \in M^2$, $wa \in L^2$, $a \in A_c$ (because we are in the second case of definition 7.8), and $P(s_j) = P(w)$. Therefore $wa \in M^2$, because M is observable with respect to L and P. Finally, in the third case we have similarly $w \in M^2$, $w_k \in M^2$, $w_k a \in M^2$, $wa \in L^2$, $a \in A_c$, and $P(w_k) = P(w)$, which gives also $wa \in M^2$. Hence $V = M_w \xrightarrow{a}$, and trivially $\langle U_a, V_a \rangle \in R$, which was to be shown. □

To illustrate the new operation, consider the following example.

Example 4. *We consider prefix-closed languages K^2 and L^2 given by the following tree automata, different from $\langle K\rangle$, resp. $\langle L\rangle$ from $\mathcal{L}$! The marked components are not considered, $A = \{a, \tau\}$, and $A_o = \{a\}$.*



Then



We have for instance $(K/^O L)_{\tau a\tau} = (K_{\tau a}/^O L_{\tau a})_\tau = K_{a\tau}/^O L_{\tau a\tau}$ according to the definition 7.8, because $K_{\tau a} \stackrel{\tau}{\not\to}$, but there exists $K_a \approx^K_{Aux} K_{\tau a}$ with $K_a \stackrel{\tau}{\to} K_{a\tau}$. Also, $K/^O L$ is indeed the infimal observable superlanguage of K as stated in theorem 7.11.

Recall that we use an order relation with respect to the second components of partial languages (Remark 7.8). As for the original definition of supervised product it can be shown in a similar way that

Theorem 7.12. *$(K/^O_U L) = \inf(\bar{C}O(K, L, P)) = \inf\{M \supseteq K :\ M$ is controllable with respect to L and A_{uc} and observable with respect to L and $P\}$. $(K/^O_U L)$ equals the infimal controllable and observable superlanguage of K.*

The proof of this theorem is similar to that of theorems 7.9 and 7.11. As a direct consequence, the supervised product is monotone with respect to the specification:

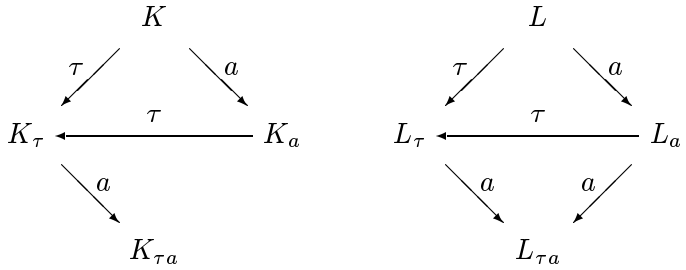
Corollary 7.13. *For (partial) languages $K \subseteq K'$ we have $(K/^O_U L) \subseteq (K'/^O_U L)$.*

Note that the infimality of the above defined operations is in both cases only with respect to the second (closed) components of the partial languages involved. The following example shows that the infimality with respect to the marking component can not hold.

Example 5. *Take* $K = (\{a\}, \{\varepsilon, a, \tau, \tau a, \tau ab\})$, $L = (\{a, ab\}, \{\varepsilon, a, ab, ab\tau, \tau, \tau a, \tau ab\})$, *and* $M = (\{a, \tau\}, \{\varepsilon, a, ab, \tau, \tau a, \tau ab\})$. *Then* $K/^O L = (\{a, ab\}, \{\varepsilon, a, ab, \tau, \tau a, \tau ab\})$. *Hence* $K \subseteq M$, M *is observable with respect to* L *and* P, *but* $(K/^O L)^1 \not\subseteq M^1$, *because* $ab \in (K/^O L)^1 \setminus M^1$.

Similar examples can be constructed for $K/_U L$ or $K/^O_U L$. Before we study the antipermissive control law, we consider the case, where controllability is again not an issue. Unlike the permissive case, the fact that $Aux(K, L)$ is not an equivalence relation on $DK \times DL$ creates difficulties as is illustrated in the example below.

Example 6. *Consider the following specification and plant languages:*



We see that $K_\tau = K_{a\tau}$ *and* $L_\tau = L_{a\tau}$. *This is a problem, because using the antipermissive control law one would like to allow* $a \in A$ *after observation of* s *if* $M = K_s$ *satisfies the condition*

$$M \xrightarrow{a} \text{ and } \forall (M', N') \in DK \times DL : \ (M', N') \approx^{K,L}_{Aux} (M, N) : \ (N' \xrightarrow{a} \Rightarrow \ M' \xrightarrow{a}). \quad (1)$$

This condition seems to be a natural coalgebraic interpretation of the antipermissive control law $\gamma_A(V, s)$ *introduced above. But the state* K_τ *can be reached by two strings, whose projections are* ε *and* a, *and this creates a difficulty. On one hand after* $s = a$, *event* a *should be disabled at* $K_\tau = K_{a\tau}$, *since* $(K_\tau, L_\tau) \approx^{K,L}_{Aux} (K_a, L_a)$ *and* $L_a \xrightarrow{a}$, *while* $K_a \not\xrightarrow{a}$. *On the other hand, after* $s = \varepsilon$, *event* a *can be enabled at* K_τ, *since the condition in the antipermissive control law* $\gamma_A(V, s)$ *is fulfilled. Using the minimal representation and condition (1) we would define by coinduction a different language than the language of the closed-loop system. The problem is that the states of the minimal representations that lie in the intersection of two observer states might lead to the conflicts as is shown in this example. In order to avoid the above ambiguities and define the closed-loop system under the antipermissive control law, suitable ("unfolded") automata representations, in general different from minimal ones, must be used.*

In order to avoid the undesirable situation of the above example we use in the following definition underlying representations of languages K and L by automata S_1 and S, where S_1 is a subautomaton of S such that $Aux(S_1)$ is an equivalence relation. We have proven in section 4 that the condition of S_1 being state-partition automaton [33] is stronger, i.e. it guarantees that $Aux(S_1)$ is an equivalence relation. It is known how to construct such representations [7] or [33].

Let s_0 denote the common initial state of S_1 and S. The transition structure of S_1 and S is denoted by $\to_1$ and $\to$, respectively. In the following algorithm we compute a sublanguage of K that is observable with respect to L and P using the antipermissive control law.

Algorithm 3. *Let automata* S_1 *and* S *represent* K *and* L, *respectively, be such that* S_1 *is a subautomaton of* S *and* $Aux(S_1)$ *is an equivalence relation. Let us construct the partial automaton* $\tilde{S} = \langle \tilde{o}, \tilde{t} \rangle$ *with the* $\tilde{t}$ *denoted by* $\to_\prime$.
1. Put $\tilde{S} := \{s_0\}$.
2. For any $s \in \tilde{S}$ *and* $a \in A$ *we put* $s \xrightarrow{a}_\prime s_a$ *if* $\forall s' \in S_1$:

$s' \approx_{Aux(S_1)} s : (s' \overset{a}{\to} \Rightarrow \ s' \overset{a}{\to}_1)$
and we put in the case $s \overset{a}{\to}'$ *also* $\tilde{S} := \tilde{S} \cup \{s_a\}$.
3. For any $s \in \tilde{S}$ *we put* $\tilde{o}(s) = o(s)$.

Let us denote by $\tilde{l}$ the unique (behavior) homomorphism given by finality of $\mathcal{L}$.

Theorem 7.14. *$\tilde{l}(s_0)$ is an observable sublanguage with respect to L and P. Moreover, if S_1 is a state-partition automaton, then $\tilde{l}(s_0)$ contains the supremal $(L,P)-$normal sublanguage of K.*

Proof. To prove the observability of $\tilde{l}(s_0)$ we show that the following relation is an observability relation on $\tilde{S} \times S$.

$$O = \{\langle (s_0)_u, (s_0)_u \rangle \mid u \in \tilde{l}(s_0)\ \}.$$

Then $\tilde{l}(s_0)$ is observable respect to L and P according to theorem 4.6. Take a pair $\langle (s_0)_v, (s_0)_v \rangle \in O$ for some $v \in \tilde{l}(s_0)$.
(i) If $(s_0)_v \overset{a}{\to}'$ for $a \in A$, then clearly by construction of Algorithm 3 $(s_0)_v \overset{a}{\to}$. It is clear from the definition of O that $\langle (s_0)_{va}, (s_0)_{va} \rangle \in O$.
(ii) Let $a \in A_c$ be such that $(s_0)_v \overset{a}{\to}$ and let there exist $s' \in \tilde{S}$: $s' \approx_{Aux(\tilde{S})} (s_0)_v$ with $s' \overset{a}{\to}$. By Lemma 4.1 there exist two strings $w, w' \in A^*$ such that $P(w) = P(w')$, $(s_0)_v = (s_0)_w$, and $s' = (s_0)_{w'} \overset{a}{\to}$. According to the construction of Algorithm 3 for any $s \approx_{Aux(S_1)} (s_0)_{w'}$ there must be $s \overset{a}{\to} \Rightarrow \ s \overset{a}{\to}_1$. In order to show that $(s_0)_v \overset{a}{\to}'$ it must be that for any $q \approx_{Aux(S_1)} (s_0)_v$ there must be $q \overset{a}{\to} \Rightarrow \ q \overset{a}{\to}_1$). But using the fact that $Aux(S_1)$ is transitive and the fact that $s' \approx_{Aux(\tilde{S})} (s_0)_v$ implies that $s' \approx_{Aux(S_1)} (s_0)_v$ we obtain that $\langle s', q \rangle \in Aux(S_1)$. But this just means that for any $q \approx_{Aux(S_1)} (s_0)_v$ we have $q \overset{a}{\to} \Rightarrow \ q \overset{a}{\to}_1$, i.e. $(s_0)_v \overset{a}{\to}'$, and O is an observability relation.

We show finally that the supremal $(L,P)-$normal sublanguage of K is contained in $\tilde{l}(s_0)$. Let N be a $(L,P)-$normal sublanguage of K. Then it is sufficient to show that

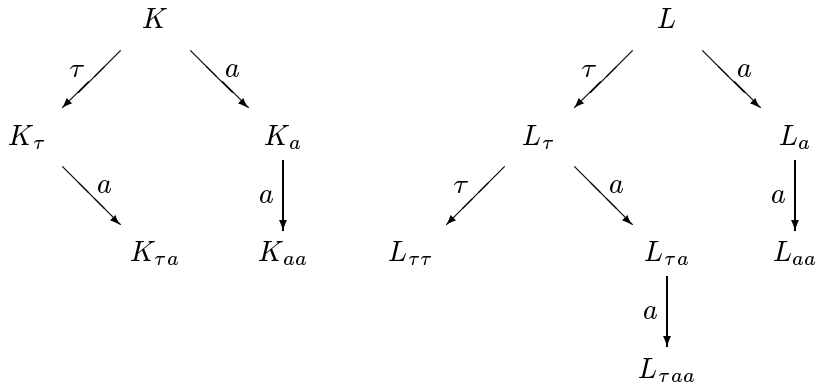
$$R = \{\langle N_u, \tilde{l}(s_0)_u \rangle \mid u \in N^2\}$$

satisfies (ii) of simulation relation in order to prove that $N^2 \subseteq \tilde{l}(s_0)^2$. Take an arbitrary pair $\langle N_w, \tilde{l}(s_0)_w \rangle \in R$ for some $w \in N^2$. Let $N_w \overset{a}{\to}$ for $a \in A$. Then also $K_w \overset{a}{\to}$, since $N \subseteq K$ and $L_w \overset{a}{\to}$ as well. This means that $(s_0)_w \overset{a}{\to}_1$ and $(s_0)_w \overset{a}{\to}$. In order to show that $\tilde{l}(s_0)_w \overset{a}{\to}$, i.e. $(s_0)_w \overset{a}{\to}'$ we must prove that for any $q \approx_{Aux(S_1)} (s_0)_w$: $q \overset{a}{\to} \ \Rightarrow \ q \overset{a}{\to}_1$. There exist $v, v' : P(v) = P(v')$ such that $q = (s_0)_{v'}$ and $(s_0)_w = (s_0)_v$. Since S_1 is a state-partition automaton and $(s_0)_w$ is in two possibly different states of the observer automaton, we conclude by the property of state-partition automaton that these two states of the observer automaton coincide. But this means that there exists $w' \in A^*$ such that $P(w) = P(w')$ and $q = (s_0)_{w'}$. Now $q \overset{a}{\to}$ means that $w'a \in L^2$. Using normality of N it follows from $wa \in N^2$ and $w'a \in L^2$ that $w'a \in N^2$. Therefore $w'a \in K^2$ (because $N \subseteq K$), which means that $q \overset{a}{\to}_1$. We conclude that $\tilde{l}(s_0)_w \overset{a}{\to}$ and R satisfies (ii) of simulation relations, i.e. we have the inclusion $N^2 \subseteq \tilde{l}(s_0)^2$. Note that since N was an arbitrary $(L,P)-$normal sublanguage of K, the same inclusion must hold for the supremal $(L,P)-$normal sublanguage of K. □

In the following example we show that $\tilde{l}(s_0)$ is not always a maximal observable sublanguage of K.

Example 7. *We consider prefix-closed languages K^2 and L^2 given again by tree automata and we assume that all the states of both automata are marked. The alphabet is $A = \{a, b, \tau\}$, with*

$A_o = \{a, b\}$.



Using algorithm 3 we obtain the resulting automaton $\tilde{S}$, whose closed language is $\tilde{l}(s_0) = \{\varepsilon, a\}$. It is indeed an observable sublanguage of K^2 containing the supremal normal sublanguage $N = \{\varepsilon\}$. However, $\tilde{l}(s_0)$ is not a maximal observable sublanguage of K^2, because $M = \{\varepsilon, a, aa\}$ is a larger observable sublanguage of K^2. Notice also that $\tilde{l}(s_0)$ is not $(L, P)-$normal.

Note that because of the above mentioned difficulties we do not present a coinductive definition of the antipermissive counterpart of supervised product that takes into account the issue of controllability, which would correspond to the definition of the antipermissive control policy. However it is possible to design an algorithm that describes the behavior of the closed-loop system under the antipermissive control policy similar to Algorithm 3. Remark that there is an asymmetry in the antipermissive control policy: it is imposed to be permissive with respect to the uncontrollable events, while it is antipermissive with respect to the controllable events. As a consequence specification K and the closed-loop language for the antipermissive control policy are not in general comparable.

Notice an important difference between the permissive and antipermissive control policy. Using the permissive control policy after having left from the specification language K by an uncontrollable event, there might still be some controllable events enabled in the future, while using the antipermissive control policy only uncontrollable events are enabled in such a situation.

To conclude, we have found an observable sublanguage that contains the supremal normal sublanguage. This is very useful, because supremal normal sublanguages are often too small (restrictive) in many concrete problems.

Our technique can be modified for constructing an observable and controllable sublanguage, because the idea in the coinductive definition of the supremal controllable sublanguage can be incorporated within Algorithm 3. In this way a monolithic algorithm for the computation of supremal normal and controllable sublanguages is developed in the next subsection.

7.3 Monolithic algorithms for supremal normal and controllable sublanguages

Now we show a monolithic algorithm for the computation of supremal normal sublanguages along the lines of Algorithm 3. The main idea is that the iterative procedure of Algorithm 1 is incorporated into Algorithm 2 using unobservable strings instead of events. Since we work with finite representations, our algorithm is still effective. Although there is already a method in the literature [11] based on the optimal control and graph theoretical techniques to obtain such a monolithic algorithm, our method is made explicitly for logical DES. The interest of this algorithm is not its computational complexity, but its formal simplicity. It is of high theoretical interest in the study

of modularly distributed DES with partial observations of local modules. It will enable us in the future [10] to find the conditions under which the global supremal normal and/or supremal normal and controllable sublanguages can be synthesized locally.

Algorithm 4. *Let automata S_1 and S representing K and L, respectively be such that S_1 is a subautomaton of S and S_1 is a state-partition automaton. Let us construct partial automaton $\tilde{S} = (\tilde{S}, \langle \tilde{o}, \tilde{t} \rangle)$ with $\tilde{t}$ denoted by $\rightarrow_!$.*
Define the auxiliary condition (*) as follows:
if $a \in A_{uo}$ then $\forall u \in A_{uo}^*$: $s_a \stackrel{u}{\rightarrow} \Rightarrow\ s_a \stackrel{u}{\rightarrow}_1$;
if $a \in A_o$ then $\forall s' \approx_{Aux(S_1)} s:\ s' \stackrel{a}{\rightarrow} \Rightarrow\ s' \stackrel{a}{\rightarrow}_1$, in which case also $\forall u \in A_{uo}^*$: $s'_a \stackrel{u}{\rightarrow} \Rightarrow\ s'_a \stackrel{u}{\rightarrow}_1$.
Below are the steps of the algorithm.
1. Put $\tilde{S} := \{s_0\}$.
2. For any $s \in \tilde{S}$ and $a \in A$ we put $s \stackrel{a}{\rightarrow}_!\ s_a$ if $s \stackrel{a}{\rightarrow}_1$ and condition (*) is satisfied and we put in the case $s \stackrel{a}{\rightarrow}_!$ also $\tilde{S} := \tilde{S} \cup \{s_a\}$.
3. For any $s \in \tilde{S}$ we put $\tilde{o}(s) = o(s)$.

Let us denote by $\tilde{l}$ the unique (behavior) homomorphism given by finality of $\mathcal{L}$.

Theorem 7.15. *$\tilde{l}(s_0)$ is the supremal $(L, P)-$normal sublanguage of K.*

Proof. To prove the normality of $\tilde{l}(s_0)$ we show that the following relation is a normal relation on $\tilde{S} \times S$.

$$N = \{\langle (s_0)_u, (s_0)_u \rangle \mid u \in \tilde{l}(s_0)^2\ \}.$$

Then $\tilde{l}(s_0)$ is $(L, P)-$normal according to Theorem 5.7. Take a pair $\langle (s_0)_v, (s_0)_v \rangle \in N$ for some $v \in \tilde{l}(s_0)^2$.
(i) If $(s_0)_v \stackrel{a}{\rightarrow}_!$ for $a \in A$, then clearly by construction of Algorithm 4 $(s_0)_v \stackrel{a}{\rightarrow}$. It is clear from the definition of N that $\langle (s_0)_{va}, (s_0)_{va} \rangle \in N$.
(ii) Let $a \in A_{uo}$ be such that $(s_0)_v \stackrel{a}{\rightarrow}$. We must show that $(s_0)_v \stackrel{a}{\rightarrow}_!$, i.e. $\forall u \in A_{uo}^*$: $(s_0)_{va} \stackrel{u}{\rightarrow} \Rightarrow\ (s_0)_{va} \stackrel{u}{\rightarrow}_1$. It follows from $(s_0) \stackrel{v}{\rightarrow}_!$ and Algorithm 4 that $\forall u \in A_{uo}^*$: $(s_0)_v \stackrel{u}{\rightarrow} \Rightarrow\ (s_0)_v \stackrel{u}{\rightarrow}_1$. Indeed, if we assume $v = v_1 \ldots v_k$ for some $k \in Z$, then either $v_k \in A_{uo}$, i.e. $(s_0)_{v_1 \ldots v_{k-1}} \stackrel{v_k}{\rightarrow}_!$ means directly that $\forall u \in A_{uo}^*$: $(s_0)_v \stackrel{u}{\rightarrow} \Rightarrow\ (s_0)_v \stackrel{u}{\rightarrow}_1$ or $v_k \in A_o$, but then the condition (*) is even stronger: by putting $s' = s$ we obtain the same conclusion. Since in both cases $au \in A_{uo}^*$, the required implication holds as well for $(s_0)_{va}$ as required for $(s_0)_v \stackrel{a}{\rightarrow}_!$.
(iii) Let $a \in A_o$ be such that $(s_0)_v \stackrel{a}{\rightarrow}$ and let there exist $s' \in \tilde{S}$: $s' \approx_{Aux(\tilde{S})} (s_0)_v$ with $s' \stackrel{a}{\rightarrow}_!$. By Lemma 4.1 there exist two strings $w, w' \in A^*$ such that $P(w) = P(w')$, $(s_0)_v = (s_0)_w$, and $s' = (s_0)_{w'} \stackrel{a}{\rightarrow}$. According to the construction of Algorithm 4 for any $s \approx_{Aux(S_1)} (s_0)_{w'}$ there must be $s \stackrel{a}{\rightarrow} \Rightarrow\ \ s \stackrel{a}{\rightarrow}_1$, in which case also $\forall u \in A_{uo}^*$: $s_a \stackrel{u}{\rightarrow} \Rightarrow\ s_a \stackrel{u}{\rightarrow}_1$. In order to show that $(s_0)_v \stackrel{a}{\rightarrow}_!$ it must be that for any $q \approx_{Aux(S_1)} (s_0)_v$ we have $q \stackrel{a}{\rightarrow} \Rightarrow\ \ q \stackrel{a}{\rightarrow}_1$, in which case also $\forall u \in A_{uo}^*$: $q_a \stackrel{u}{\rightarrow} \Rightarrow\ \ q_a \stackrel{u}{\rightarrow}_1$. But using the fact that $Aux(S_1)$ is transitive, because S_1 is a state-partition automaton, a stronger condition, and the fact that $s' \approx_{Aux(\tilde{S})} (s_0)_v$ implies that $s' \approx_{Aux(S_1)} (s_0)_v$ we obtain that $\langle s', q \rangle \in Aux(S_1)$. But this just means that for any $q \approx_{Aux(S_1)} (s_0)_v$ we have $q \stackrel{a}{\rightarrow} \Rightarrow\ \ q \stackrel{a}{\rightarrow}_1$, in which case also $\forall u \in A_{uo}^*$: $q_a \stackrel{u}{\rightarrow} \Rightarrow\ q_a \stackrel{u}{\rightarrow}_1$, i.e. $(s_0)_v \stackrel{a}{\rightarrow}_!$. Therefore N is a normal relation.

We show finally that the supremal $(L, P)-$normal sublanguage of K is contained in $\tilde{l}(s_0)$. Let N be a $(L, P)-$normal sublanguage of K. Then it is sufficient to show that

$$R = \{\langle N_u, \tilde{l}(s_0)_u \rangle \mid u \in N^2\}$$

satisfies (ii) of simulation relation in order to prove that $N^2 \subseteq \tilde{l}(s_0)^2$. Take an arbitrary pair $\langle N_w, \tilde{l}(s_0)_w \rangle \in R$ for some $w \in N^2$. Let $N_w \xrightarrow{a}$ for $a \in A$. Then also $K_w \xrightarrow{a}$, since $N \subseteq K$ and $L_w \xrightarrow{a}$ as well. This means that $(s_0)_w \xrightarrow{a}_1$ and $(s_0)_w \xrightarrow{a}$. In order to show that $\tilde{l}(s_0)_w \xrightarrow{a}$, i.e. $(s_0)_w \xrightarrow{a}_\prime$ it must be shown that the condition (*) is satisfied.

For $a \in A_{uo}$ we need to show that $\forall u \in A^*_{uo}$: $(s_0)_{wa} \xrightarrow{u} \Rightarrow \ (s_0)_{wa} \xrightarrow{u}_1$. But this is easy: $(s_0)_{wa} \xrightarrow{u}$ means $wau \in L^2$. Since N is $(L,P)-$normal, $wa \in N^2$ and $P(wa) = P(wau)$, we deduce $wau \in N^2 \subseteq K^2$. But this just means that $(s_0)_{wa} \xrightarrow{u}_1$.

For $a \in A_o$ it must be checked that for any $q \approx_{Aux(S_1)} (s_0)_w$: $q \xrightarrow{a} \ \Rightarrow \ q \xrightarrow{a}_1$, in which case also $\forall u \in A^*_{uo}$: $q_a \xrightarrow{u} \Rightarrow \ q_a \xrightarrow{u}_1$. There exist $v, v' : \ P(v) = P(v')$ such that $q = (s_0)_{v'}$ and $(s_0)_w = (s_0)_v$. Since S_1 is a state-partition automaton and $(s_0)_w = (s_0)_v$ is in two potentially different states of the observer automaton, we conclude by the property of state-partition automaton that these two states of the observer automaton coincide. But this means that there exists $w' \in A^*$ such that $P(w) = P(w')$ and $q = (s_0)_{w'}$. Now $q \xrightarrow{a}$ means that $w'a \in L^2$. By normality of N it follows from $wa \in N^2$ and $w'a \in L^2$ that $w'a \in N^2$. Therefore $w'a \in K^2$ (because $N \subseteq K$), which means that $q \xrightarrow{a}_1$. The rest is similar as for $a \in A_{uo}$: if for $u \in A^*_{uo}$: $q_a = (s_0)_{w'a} \xrightarrow{u}$, then $w'au \in L^2$, by normality of N and using $wa \in N^2$, where $P(w'au) = P(wa)$ we have $w'au \in N^2 \subseteq K^2$. But this just means that $(s_0)_{w'a} = q_a \xrightarrow{u}_1$.

We conclude that $\tilde{l}(s_0)_w \xrightarrow{a}$ and R satisfies (ii) of simulation relation, i.e. we have the inclusion $N^2 \subseteq \tilde{l}(s_0)^2$. Note that since N was arbitrary $(L,P)-$normal sublanguage of K, and $\tilde{l}(s_0)$ has been shown to be a $(L,P)-$ normal sublanguage of K, it follows that $\tilde{l}(s_0)$ is the supremal $(L,P)-$normal sublanguage of K. □

Following the same technique we can synthesize a monolithic algorithm for computation of supremal normal and controllable sublanguages.

Algorithm 5. *Let automata S_1 and S representing K and L, respectively are such that S_1 is a subautomaton of S and S_1 is a state-partition automaton. Let us construct partial automaton $\tilde{S} = (\tilde{S}, \langle \tilde{o}, \tilde{t} \rangle)$ with $\tilde{t}$ denoted by $\rightarrow_\prime$.*
Define the auxiliary condition (**) as follows:
if $a \in A_u \cup A_{uo}$ then $\forall u \in (A_u \cup A_{uo})^*$: $s_a \xrightarrow{u} \Rightarrow \ s_a \xrightarrow{u}_1$;
if $a \in A_c \cap A_o$ then $\forall s' \approx_{Aux(S_1)} s : \ s' \xrightarrow{a} \Rightarrow \ s' \xrightarrow{a}_1$, in which case also $\forall u \in (A_u \cup A_{uo})^*$: $s'_a \xrightarrow{u} \Rightarrow \ s'_a \xrightarrow{u}_1$.

Below are the steps of the algorithm.
1. Put $\tilde{S} := \{s_0\}$.
2. For any $s \in \tilde{S}$ and $a \in A$ we put $s \xrightarrow{a}_\prime s_a$ if $s \xrightarrow{a}_1$ and condition (**) is satisfied and we put in the case $s \xrightarrow{a}_\prime$ also $\tilde{S} := \tilde{S} \cup \{s_a\}$.
3. For any $s \in \tilde{S}$ we put $\tilde{o}(s) = o(s)$.

As usual, we denote by $\tilde{l}$ the unique (behavior) homomorphism given by finality of $\mathcal{L}$. Similarly as for Algorithm 4, one can verify by coinduction that

Theorem 7.16. *$\tilde{l}(s_0)$ is the supremal controllable (with respect to L and A_u) and $(L,P)-$normal sublanguage of K.*

Proof. The structure of the proof follows very much that of Theorem 7.15. First we prove that $\tilde{l}(s_0)$ is controllable with respect to L and A_u. According to Theorem 5.8 it is sufficient to show that the following relation is a control relation on $\tilde{S} \times S$.

$$C = \{\langle (s_0)_u, (s_0)_u \rangle \mid u \in \tilde{l}(s_0)^2 \ \}.$$

Take a pair $\langle (s_0)_w, (s_0)_w \rangle \in N$ for some $w \in \tilde{l}(s_0)^2$.
(i) If $(s_0)_w \stackrel{a}{\rightarrow}_\prime$ for $a \in A$, then clearly by construction of Algorithm 5 $(s_0)_w \stackrel{a}{\rightarrow}$, because $\tilde{l}(s_0) \subseteq K \subseteq L$. It is clear from the definition of C that $\langle (s_0)_{wa}, (s_0)_{wa} \rangle \in C$.
(ii) Let $a \in A_u$ be such that $(s_0)_w \stackrel{a}{\rightarrow}$. We must show that $(s_0)_w \stackrel{a}{\rightarrow}_\prime$. According to Algorithm 5 condition (**) must be checked. Since $A_u \subseteq A_u \cup A_{uo}$ it amounts to show that $\forall u \in (A_u \cup A_{uo})^*$: $(s_0)_{wa} \stackrel{u}{\rightarrow} \Rightarrow \ (s_0)_{wa} \stackrel{u}{\rightarrow}_1$. We have $u = u_1 \ldots u_l$ for some $l \in Z$ with $\forall i \in \{1, \ldots, l\} : \ u_i \in A_u \cup A_{uo}$. Notice that we have also $w = w_1 \ldots w_k$ for some $k \in Z$. There are 2 possibilities for w_k: $w_k \in A_u \cup A_{uo}$ or $w_k \in A_c \cap A_o$. According to condition (**) of Algorithm 5 for $(s_0) \stackrel{w}{\rightarrow}_\prime$, i.e. $(s_0)_{w_1 \ldots w_{k-1}} \stackrel{w_k}{\rightarrow}_\prime$ in both cases means that in particular $\forall u \in (A_u \cup A_{uo})^*$: $s_w \stackrel{u}{\rightarrow} \Rightarrow \ s_w \stackrel{u}{\rightarrow}_1$. Indeed, condition (**) for $w_k \in A_c \cap A_o$ is stronger than for $w_k \in A_u \cup A_{uo}$ as is easily seen by taking $s' = s$. Since $au \in (A_u \cup A_{uo})^*$, the condition (**) for $(s_0)_w \stackrel{a}{\rightarrow}_\prime$ holds true, which proves the controllability of $\tilde{l}(s_0)$.

To prove the normality of $\tilde{l}(s_0)$ we show that the following relation is a normal relation on $\tilde{S} \times S$.

$$N = \{ \langle (s_0)_u, (s_0)_u \rangle \mid u \in \tilde{l}(s_0)^2 \ \}.$$

Then $\tilde{l}(s_0)$ is normal with respect to L and P according to Theorem 5.7. Take a pair $\langle (s_0)_v, (s_0)_v \rangle \in N$ for some $v \in \tilde{l}(s_0)^2$.
(i) If $(s_0)_v \stackrel{a}{\rightarrow}_\prime$ for $a \in A$, then clearly by construction of Algorithm 5 $(s_0)_v \stackrel{a}{\rightarrow}$. It is clear from the definition of N that $\langle (s_0)_{va}, (s_0)_{va} \rangle \in N$.
(ii) Let $a \in A_{uo}$ be such that $(s_0)_v \stackrel{a}{\rightarrow}$. We must show that $(s_0)_v \stackrel{a}{\rightarrow}_\prime$, i.e. $\forall u \in (A_u \cup A_{uo})^*$: $(s_0)_{va} \stackrel{u}{\rightarrow} \Rightarrow \ (s_0)_{va} \stackrel{u}{\rightarrow}_1$. It follows from $(s_0) \stackrel{v}{\rightarrow}_\prime$, $A_{uo} \subseteq A_u \cup A_{uo}$, and Algorithm 5 that $\forall u \in (A_u \cup A_{uo})^*$: $(s_0)_v \stackrel{u}{\rightarrow} \Rightarrow \ (s_0)_v \stackrel{u}{\rightarrow}_1$, the argument being the same as above in the proof of controllability. Since $au \in (A_u \cup A_{uo})^*$, the required implication holds as well.
(iii) Let $a \in A_o$ be such that $(s_0)_v \stackrel{a}{\rightarrow}$ and let there exists $s' \in \tilde{S}$: $s' \approx_{Aux(\tilde{S})} (s_0)_v$ with $s' \stackrel{a}{\rightarrow}_\prime$. By Lemma 4.1 there exist two strings $w, w' \in A^*$ such that $P(w) = P(w')$, $(s_0)_v = (s_0)_w$, and $s' = (s_0)_{w'} \stackrel{a}{\rightarrow}_\prime$. But using the fact that S_1 is a state-partition automaton there exists $v' : \ P(v') = P(v)$ such that $s' = (s_0)_{v'} \stackrel{a}{\rightarrow}_\prime$. Two cases must be distinguished. Assume first that $a \in A_o \cap A_c$. It follows from Algorithm 5 that $\forall s \approx_{Aux(S_1)} (s_0)_{v'}$ there must be $s \stackrel{a}{\rightarrow} \Rightarrow \ s \stackrel{a}{\rightarrow}_1$, in which case also $\forall u \in (A_u \cup A_{uo})^*$: $s_a \stackrel{u}{\rightarrow} \Rightarrow \ s_a \stackrel{u}{\rightarrow}_1$. Then $s \stackrel{a}{\rightarrow}_\prime$ as well using the transitivity of $Aux(S_1)$ and the obvious fact that $Aux(\tilde{S}) \subseteq Aux(S_1)$, which means that $s' \approx_{Aux(\tilde{S})} (s_0)_v$ implies that $s' \approx_{Aux(S_1)} (s_0)_v$. Now for any $q \approx_{Aux(S_1)} (s_0)_v$ there must be $\langle s', q \rangle \in Aux(S_1)$. Therefore $q \stackrel{a}{\rightarrow} \Rightarrow \ q \stackrel{a}{\rightarrow}_1$, in which case also $\forall u \in (A_u \cup A_{uo})^*$: $q_a \stackrel{u}{\rightarrow} \Rightarrow \ q_a \stackrel{u}{\rightarrow}_1$. Thus $(s_0)_v \stackrel{a}{\rightarrow}_\prime$. Now we assume that $a \in A_o \cap A_u$. According to the construction of Algorithm 5, it is sufficient to show that $\forall u \in (A_u \cup A_{uo})^*$: $(s_0)_{va} \stackrel{u}{\rightarrow} \Rightarrow \ (s_0)_{va} \stackrel{u}{\rightarrow}_1$. We know that $(s_0) \stackrel{v}{\rightarrow}_\prime$. Using the same argument as in the proof of controllability or (ii) of normality it follows that $\forall u \in (A_u \cup A_{uo})^*$: $(s_0)_v \stackrel{u}{\rightarrow} \Rightarrow \ (s_0)_v \stackrel{u}{\rightarrow}_1$. It is sufficient to notice that $a \in A_o \cap A_u \subseteq A_u \cup A_{uo}$, i.e. also $au \in (A_u \cup A_{uo})^*$. Thus, $\forall u \in (A_u \cup A_{uo})^*$: $(s_0)_{va} \stackrel{u}{\rightarrow} \Rightarrow \ (s_0)_v \stackrel{au}{\rightarrow} \Rightarrow \ (s_0)_v \stackrel{au}{\rightarrow}_1$, which is equivalent to $(s_0)_{va} \stackrel{u}{\rightarrow}_1$. Since in both cases $(s_0)_v \stackrel{a}{\rightarrow}_\prime$, we conclude that N is a normal relation.

We show finally that the supremal controllable (with respect to L and A_u) and $(L, P)-$normal sublanguage of K is contained in $\tilde{l}(s_0)$. Let N be a controllable and $(L, P)-$normal sublanguage of K. Then it is sufficient to show that

$$R = \{ \langle N_u, \tilde{l}(s_0)_u \rangle \mid u \in N^2 \}$$

satisfies (ii) of simulation relation in order to prove that $N^2 \subseteq \tilde{l}(s_0)^2$. Take an arbitrary pair $\langle N_w, \tilde{l}(s_0)_w \rangle \in R$ for some $w \in N^2$. Let $N_w \stackrel{a}{\rightarrow}$ for $a \in A$. Then also $K_w \stackrel{a}{\rightarrow}$, since $N \subseteq K$ and $L_w \stackrel{a}{\rightarrow}$ as well. This means that $(s_0)_w \stackrel{a}{\rightarrow}_1$ and $(s_0)_w \stackrel{a}{\rightarrow}$. In order to show that $\tilde{l}(s_0)_w \stackrel{a}{\rightarrow}$, i.e.

$(s_0)_w \overset{a}{\rightarrow}'$ it must be shown that condition (**) of Algorithm 5 is satisfied. If $a \in A_u \cup A_{uo}$ then we show that $\forall u \in (A_u \cup A_{uo})^*$: $(s_0)_{wa} \overset{u}{\rightarrow} \Rightarrow \ (s_0)_{wa} \overset{u}{\rightarrow}_1$. Indeed, if $u \in (A_u \cup A_{uo})^*$ then $u = u_1 \dots u_k$ for some $k \in Z$ with $u_i \in A_u \cup A_{uo}$ $\forall i \in \{1, \dots, k\}$. Thus, $(s_0)_{wa} \overset{u}{\rightarrow}$, i.e. $wau = wau_1 \dots u_k \in L^2$ together with $wa \in N^2$ and normality and controllability of N inductively used implies $wau_1 \in N^2$,...,$wau = wau_1 \dots u_k \in N^2 \subseteq K^2$. This means that $(s_0)_{wa} \overset{u}{\rightarrow}_1$, which was to be shown. Let $a \in A_c \cap A_o$. We need to show that $\forall s' \approx_{Aux(S_1)} (s_0)_w : \ s' \overset{a}{\rightarrow} \Rightarrow \ s' \overset{a}{\rightarrow}_1$, in which case also $\forall u \in (A_u \cup A_{uo})^*$: $s'_a \overset{u}{\rightarrow} \Rightarrow \ s'_a \overset{u}{\rightarrow}_1$. Let $s' \approx_{Aux(S_1)} (s_0)_w : \ s' \overset{a}{\rightarrow}$. According to Lemma 4.1 and by taking into account that S_1 is a state-partition automaton, there exists $w' \in K^2$ such that $P(w') = P(w)$ and $s' = (s_0)_{w'}$. Hence $s' \overset{a}{\rightarrow}$ is equivalent to $w'a \in L^2$. By normality of N it follows from $wa \in N^2$ and $w'a \in L^2$ that $w'a \in N^2$. Thus $w'a \in K^2$, because $N^2 \subseteq K^2$, but this means that $s' \overset{a}{\rightarrow}_1$. The second part is similar as for $a \in A_{uo} \cup A_u$. Indeed, for $u \in (A_u \cup A_{uo})^*$ with $s'_a \overset{u}{\rightarrow}$ we obtain consequently: $w'au \in L^2$, $w'a \in N^2$, i.e. by inductive application of normality and controllability of N we have finally $w'au \in N^2 \subseteq K^2$, which gives $s'_a \overset{u}{\rightarrow}_1$. To conclude, in any case we have obtained $\tilde{l}(s_0)_w \overset{a}{\rightarrow}$, i.e. R satisfies (ii) of simulation relation, and the inclusion $N^2 \subseteq \tilde{l}(s_0)^2$ has been shown. Note that since N was arbitrary controllable and $(L, P)-$normal sublanguage of K, it follows that $\tilde{l}(s_0)$ is the supremal controllable and $(L, P)-$normal sublanguage of K. □

Remark 7.17. *An important feature of Algorithm 5 is its compactness, i.e. it is not an iteration of two separate algorithms as are the algorithms in [33] or [9]. Therefore it looks almost like a coinductive definition of the supremal normal and controllable sublanguage, which is not possible to do directly in $\mathcal{L}$. Thus Algorithm 5 is suitable for investigating problems like "when does the supremal normal and controllable sublanguage commute with the synchronous product of (partial) languages?"*

7.4 Distributivity of the supervised product

The behavior of the supervised DES has been formalized by the (partial) language operation of supervised product. It is of interest to study algebraic properties of this operation, e.g. distributivity with respect to (partial) language operations. The problem of distributivity of the supervised product with respect to language unions is addressed in this section. The following theorem answers the main question. It turns out that

Theorem 7.18. *If $A_c \subseteq A_o$, then for any K and K' (partial) sublanguages of L we have:*

$$(K \cup K')/_U^O L = (K/_U^O L) \cup (K'/_U^O L).$$

Proof. Formally, it can be checked that

$$R = \{\langle [(K \cup K')/_U^O L]_u, [(K/_U^O L) \cup (K'/_U^O L)]_u \rangle \mid u \in [(K \cup K')/_U^O L]^2\}$$

is a bisimulation relation. Take a $w \in [(K \cup K')/_U^O L]^2$.
(i) is straightforward: $[(K \cup K')/_U^O L]_w \downarrow$ iff $w \in L^1$ iff $[(K/_U^O L) \cup (K'/_U^O L)]_w \downarrow$.
(ii) If $[(K \cup K')/_U^O L]_w \overset{a}{\rightarrow}$ for $a \in A$, then according to the definition 7.4 of supervised products

several cases must be distinguished. We have the following possibilities:

$$[(K \cup K')/^O_U L]_w = \begin{cases} (K \cup K')_w/^O_U L_w & \text{if } (K \cup K') \overset{w}{\to} \text{ and } L \overset{w}{\to} \\ \cup_{i \in I}(K \cup K')_{w_i}/^O_U L_w & \text{if } (K \cup K') \overset{w}{\not\to} \text{ and } L \overset{w}{\to} \text{ and } \exists I \neq \emptyset \\ & \text{and } w_i,\ i \in I:\ P(w_i) = P(w) \text{ and } \forall i \in I: \\ & (K \cup K') \overset{w_i}{\to} \\ 0/^O_U L_w & \text{if } K \overset{w}{\not\to} \text{ and } L \overset{w}{\to} \text{ and } w \notin A_c^* \end{cases}$$

By applying again the definition of the supervised product several cases must be distinguished:

$$[(K \cup K')/^O_U L]_{wa} = \begin{cases} (K \cup K')_{wa}/^O_U L_{wa} & \text{if } (K \cup K') \overset{wa}{\to} \text{ and } L \overset{wa}{\to} \\ \cup_{j \in J}(K \cup K')_{v_j}/^O_U L_{wa} & \text{if } (K \cup K') \overset{w}{\not\to} \text{ and } L \overset{wa}{\to} \text{ and } a \in A_c \cup A_o \\ & \text{and } \exists J \neq \emptyset \text{ and } v_j,\ j \in J:\ P(v_j) = P(w) \\ & \text{and } \forall j \in J:\ (K \cup K')_{v_j} \overset{a}{\to} \\ (K \cup K')_w/^O_U L_{wa} & \text{if } K \overset{wa}{\not\to} \text{ and } L \overset{wa}{\to} \text{ and } a \in A_{uc} \cap A_{uo} \\ 0/^O_U L_{wa} & \text{if } K \overset{wa}{\not\to} \text{ and } L \overset{wa}{\to} \text{ and } \forall v:\ P(v) = P(w): \\ & K_v \overset{a}{\not\to} \text{ and } a \in A_{uc} \cap A_o \end{cases}$$

Now combinations of different cases of both preceding equations must be considered. Some of the combinations are only hypothetic, and in fact they are impossible. For instance, if the last case in the first equation occurs, then only the last case in the second equation can occur. Some cases are easy, others are problematic. For instance, the last case of the other equation in combination with any case of the first equation, as well as the combination of the first cases of both equations are not problematic and the conclusion is easily drawn. Now we consider the problematic case $(K \cup K') \overset{w}{\to}$, namely e.g. $K \overset{w}{\to}$ and $K' \overset{w}{\not\to}$, while there exists an index set J such that $\forall j \in J$: $(K \cup K')_{v_j} \overset{a}{\to}$ and $P(v_j) = P(w)$, namely e.g. $\forall j \in J : K_{v_j} \overset{a}{\not\to}$ and $K'_{v_j} \overset{a}{\to}$. This is a problem, because in order to draw the plausible conclusion $(K'/^O_U L)_w \overset{a}{\to}$, we need first be sure that $(K'/^O_U L) \overset{w}{\to}$, which is not obvious. However our assumption $A_c \subseteq A_o$ will be used. It is known from Corollary 5.9 that under the assumption $A_c \subseteq A_o$ observability together with controllability are equivalent to the normality. Since the supervised product is known to be controllable and observable, it follows that the supervised product is also $(L, P)-$normal. Therefore $(K'/^O_U L)$ is $(L, P)-$ normal and from $K'_{v_j} \overset{a}{\to}$, i.e. $(K'/^O_U L) \overset{v_j}{\to}$ it follows that $(K'/^O_U L) \overset{w}{\to}$, and thus $(K \cup K'/^O_U L)_w \overset{a}{\to}$.

The same problem appears if the second case in the first equation occurs. The situation is similar to the one above with w replaced by w_i, but owing to the $(L, P)-$ normality of the supervised product this is not substantial: again $(K'/^O_U L) \overset{v_j}{\to}$ implies that $(K'/^O_U L) \overset{w}{\to}$.
(iii) This inclusion (simulation) is easy and holds always: it follows from the monotonicity of the supervised product with respect to the specification (see Corollary 7.13), therefore $K \subseteq K \cup K'$ implies that $K/^O_U L \subseteq (K \cup K')/^O_U L$. Similarly, $K' \subseteq K \cup K'$ implies that $K'/^O_U L \subseteq (K \cup K')/^O_U L$. Hence, $(K/^O_U L) \cup (K'/^O_U L \subseteq (K \cup K')/^O_U L$.

□

Under the structural assumption on the event set $A_c \subseteq A_o$, the supervised product with partial observations distributes with laguage unions. This distributivity implies that important properties are preserved by unions: if $K/^O_U L = K$, i.e. K is controllable, $L_m(G)-$closed

and observable and $K'/^O_U L = K$, i.e. K' is controllable, $L_m(G)-$closed and observable, then $(K \cup K')/^O_U L = (K/^O_U L) \cup (K'/^O_U L) = K \cup K'$, i.e. $K \cup K'$ is also controllable, $L_m(G)-$closed and observable. Note finally that it is not difficult to extend the above distributivity to an arbitrary number of specifications, even to an infinite number (which amounts to the lattice theoretical lower semicontinuity).

Similarly, if $A_c \subseteq A_o$ then our technique can be used to show that the supervised product is also distributive with respect to partial language intersections. The situation is symmetric in the sense that the opposite inclusion is trivial here (always holds). To conclude, we have shown that the concept of supervised product is useful for investigation of properties of closed-loop languages in discrete-event control with partial observations.

8 Conclusion

Supervisory control of DES with partial observations has been treated by coalgebraic techniques. The new concept of deterministic weak transitions gives rise to the definition of projected and observer automata. Observability and normality have been characterized by appropriate relations in this framework, which gives an insight into problems of partially observed DES. They have been used to design algorithms for supremal normal and/or normal and controllable sublanguages. These are discussed in detail and compared to those encountered in the literature.

Another approach, based on finality of the automaton of partial languages, consists in using coinductive or similar definitions for describing permissive or antipermissive control laws under partial observations. As a byproduct coinductive definitions of observable approximations of a given language have been obtained. These definitions give rise to new algorithms for the computation of infimal closed and observable superlanguages and observable sublanguages larger than the supremal normal sublanguage because of their coinductive nature. They rely only on observational indistinguishability relations, which can be constructed directly from the corresponding definitions that give at the same time algorithms for their construction. The lack of the existence of an optimal (maximally permissive) solution for the supervisory control with partial observations is related to the fact that the supervised product does not in general distribute with (partial) language unions when the controller has only partial information about the DES.

The naturally algorithmic character of the coalgebraic approach is one of its main advantages. While the algebraic approach works with strings (words), which is sometimes cumbersome, the coalgebraic approach relies on the relational framework (various weakening of bisimulation relations) and we proceed event by event. The use of coinductive definitions and proofs makes coalgebraic techniques relevant for control of DES. Coinductive definitions enable to characterize languages of the closed (controlled) DES and coinductive proofs are used to check different properties like controllability, observability, normality, or distributivity of operations on (partial) languages.

The results of this paper are being generalized to the decentralized and modular supervisory control. For instance, in modular control of DES, the system is composed of local subsystems that run concurrently, i.e. the global system is the parallel composition of local systems. To each local system a local supervisor is associated. Many interesting questions arise: can the control be exerted at the local level without violating our control objectives or without affecting the optimality of the solution? If the answer to these question is positive, there is an exponential save on the computational complexity. In our coalgebraic framework these two problems can be paraphrased as follows: when does the supervised product commute with synchronous product

and when does the supremal controllable sublanguage (as an operation defined by coinduction in section 7) commute with synchronous product? (recall that the synchronous product of partial languages has been defined by coinduction in [20]).

The contribution of the coalgebraic approach to the control and systems theory remains to be further evaluated. However, we believe that application of the coinductive techniques is not limited to discrete-event systems, but it can be useful for other type of systems. It seems possible to study with coalgebraic techniques some problems of hybrid systems, especially if the control objectives are only at the discrete-event level (safety or minimal required behavior). In some areas of control and systems theory with a high level of abstraction there might be interesting to apply the coalgebraic techniques. Various coalgebras (systems) can be obtained by varying the functor on the category of sets. Moreover, the use of this method is not limited to systems defined by functors in the category of sets, but functors on some "structurally richer" categories (like the categories of topological or metric spaces and continuous functions between them as morphisms). An interesting application of coalgebra to symbolic dynamics in one dimensional discrete-time dynamical systems defined by a continuous function on a complete metric space can be found in [21]. Different types of coalgebras have their own notions of homomorphism and bisimulation, as well as cofreeness and finality, i.e. coinduction, yet there is a unifying theory of universal coalgebra.

Future research tasks might include a study of how to improve the computational complexity of our algorithms, a study of decentralized, hierarchical, and modular control of DES using coalgebra as well as an application of the coalgebraic techniques to timed DES. Optimal supervisory control can be investigated by coalgebraic techniques using coalgebras of weighted automata [22] (weights here correspond to the costs) with formal power series as their final coalgebra.

Acknowledgement

The investigation was primarily carried out during the first author's stay at CWI Amsterdam, where he has worked on the NWO project COCON together with his research advisor Jan H. van Schuppen. The discussion with Stéphane Lafortune and his Ph.D. students is gratefully acknowledged and has motivated a part of the results of section 7.

References

[1] A. Bergeron. A Unified Approach to Control Problems in Discrete Event Processes. *Informatique Théorique et Applications*, Volume 27, 6, pp. 555-573, 1993.

[2] G. Barrett and S. Lafortune. Bisimulation, the Supervisory Control Problem and Strong Model Matching for Finite State Machines. *Journal of Discrete Event Dynamical Systems: Theory and Application* Vol. 8, No. 4, pp. 377-429, 1998.

[3] R.D. Brandt, V. Garg, R. Kumar, F. Lin, S.I. Marcus, W.M. Wonham. Formulas for Calculating Supremal Controllable and Normal Sublanguages, *Systems & Control Letters* 15: 111-117, 1990.

[4] R. Cieslak, C. Desclaux, A.Fawaz, and P. Varayia. Supervisory Control of a Class of Discrete Event Processes. *IEEE Trans. Automatic Control*, 33:249-260, 1988.

[5] S.G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*, Kluwer Academic Publishers, Dordrecht 1999.

[6] H. Cho and S. I. Marcus. On Supremal Languages of Classes of Sublanguages that Arise in Supervisor Synthesis Problems with Partial Observations. *Mathematics of Control, Signal, and Systems*, 2:47-69, 1989.

[7] H. Cho and S. I. Marcus. Supremal and Maximal Sublanguages Arising in Supervisor Synthesis Problems with Partial Observations. *Math. Systems Theory*, 22:171-211, 1989.

[8] J. Komenda. Computation of Supremal Sublanguages of Supervisory Control Using Coalgebra. Proceedings *WODES'02*, Workshop on Discrete-Event Systems, Zaragoza, p. 26-33, October 2-4, 2002.

[9] J. Komenda. Coalgebra and Supervisory Control of Discrete-Event Systems with Partial Observations. Proceedings of *MTNS 2002*, Notre Dame (IN), August 2002.

[10] J. Komenda ad J.H. van Schuppen. Supremal Normal Sublanguages of Large Distributed Discrete-Event Systems. Proceedings *WODES'04*, Workshop on Discrete-Event Systems, Reims, September 22-24, 2004.

[11] R. Kumar, V. K. Garg. Optimal supervisory control of discrete event dynamical systems, SIAM J. Control and Optimization 33, 419-439, 1995.

[12] R. Kumar, M. Heymann. Masked prioritized synchronization for interaction and control of discrete event systems, IEEE Transaction Automatic Control 45, 1970-1982, 2000.

[13] S. Lafortune and E. Chen. The Infimal Closed and Controllable Superlanguage and its Applications in Supervisory Control, *IEEE Trans. on Automatic Control*, Vol. 35,N 4, p. 398-405, 1990.

[14] F. Lin and W.M. Wonham, On Observability of Discrete-Event Systems, *Information Sciences*,44: 173-198, 1988.

[15] F. Lin and W.M. Wonham, Decentralized Control and Coordination of Discrete-Event Systems with Partial Observations, *IEEE Trans. Automatic Control*, 35:1330-1337, 1990.

[16] R. Milner. A complete aximatisation for observational congruence of finite-state behaviors. *Inform. and Computation*, 81, 227-247, 1989.

[17] R. Milner. Communication and Concurrency. *Prentice Hall International Series in Computer Science.* Prentice Hall International, New York, 1989.

[18] A. Overkamp and J.H. van Schuppen. Maximal Solutions in Decentralized Supervisory Control, *SIAM Journal on Control and Optimization*, Vol. 39, No.2, pp. 492-511, 2000.

[19] J.J.M.M. Rutten. Automata and Coinduction (an Exercise in Coalgebra). *Research Report CWI*, SEN-R9803, Amsterdam, May 1998. Available also at `http://www.cwi.nl/~janr`.

[20] J.J.M.M. Rutten. Coalgebra, Concurrency, and Control. *Research Report CWI*, SEN-R9921, Amsterdam, November 1999. Available also at `http://www.cwi.nl/~janr`.

[21] J.J.M.M. Rutten. Universal Coalgebra: A Theory of Systems. *Theoretical Computer Science* 249(1):3-80, 2000.

[22] J.J.M.M. Rutten. Fundamental Study. Behavioural Differential Equations: a Coinductive Calculus of Streams, Automata, and Power Series. *Theoretical Computer Science* 308(1):1-53, 2003.

[23] P.J. Ramadge and W.M. Wonham. The Control of Discrete-Event Systems. *Proc. IEEE,* 77:81-98, 1989.

[24] K. Rudie and W.M. Wonham. The Infimal Prefix-Closed and Observable Superlanguage of a Given Language. *Systems & Control Letters* 15 (1990), 361-371.

[25] K. Rudie and W.M. Wonham. Think Globally, Act Locally: Decentralized Supervisory Control, *IEEE Trans. on Automatic Control*, Vol. 37,N 11, p. 1692-1708, 1992.

[26] A. Tarski. A lattice-theoretical fixpoint theorem and its applications, *Pacific Journal of Mathematics*, 5:285–309, 1955.

[27] S. Takai and T. Ushio. Effective Computation of an $L_m(G)$-closed, Controllable, and Observable Sublanguage Arising in Supervisory Control. Proceedings *WODES'02*, Workshop on Discrete-Event Systems, Zaragoza, p.34-39, October 2-4, 2002.

[28] J.G. Thistle. Supervisory control of discrete event systems. *Mathematical and Computer Modeling*, 11/12, pp. 25-53, 1996.

[29] J.G. Thistle and W.M. Wonham. Supervision of infinite behavior of discrete-event systems, *SIAM Journal on Control and Optimization* 32 (4), pp. 1098-1113, 1994.

[30] J.N. Tsitsiklis. On the Control of Discrete-Event Dynamical Systems, *Mathematics of Control, Signal, and Systems*, 95-107, 1989.

[31] W.M. Wonham. Towards an abstract internal model principle, *IEEE Transactions on Systems, Man and Cybernetics, SMC*, 6 (11), pp. 735-740, 1976.

[32] W.M. Wonham and P.J. Ramadge. On the Supremal Controllable Sublanguage of a Given Language, *SIAM J. Control Optim.*, 25:637-659, 1987.

[33] T.S. Yoo, S. Lafortune, and F.Lin. A Uniform Approach for Computing Supremal Sublanguages Arising in Supervisory Control theory. *Preprint, Dept. of Electrical Engineering and Computer Science, University of Michigan*, Ann Arbor 2001.

[34] T.S. Yoo, S. Lafortune. *General Architecture for Decentralized Supervisory Control of Discrete-Event Systems*. Discrete Event Dynamic Systems: Theory and Applications, 12, 335-377, 2002.