ON THE COMPOSITION OF CONDITION/EVENT SYSTEMS

S. Kowalewski[1], Y. Lakhnech[2], B. Lukoschus[2], L. Urbina[2]

[1] Lehrstuhl für Anlagensteuerungstechnik, Fachbereich Chemietechnik, Universität Dortmund, Emil-Figge-Str. 70, D-44221 Dortmund, Germany. Email: stefan.kowalewski@ast.chemietechnik.uni-dortmund.de
[2] Institut für Informatik und Praktische Mathematik der Christian-Albrechts-Universität zu Kiel, Preußerstr. 1–9, D-24105 Kiel, Germany. Email: {yl,bls,lu}@informatik.uni-kiel.de

Abstract

In this paper, several aspects of the composition of discrete state condition/event systems according to Sreenivas and Krogh are discussed. We introduce a new composition operator called parallel interconnection. It is shown to be a generalization of the originally defined cascade and feedback operators. Based on this composition operator, a modular refinement rule is presented which can be used to prove that a parallel interconnection of condition/event systems refines a given other one.

Keywords
Condition/event systems, composition, modularity, refinement.

1 Introduction

Any modular modeling approach as the one established by the condition/event systems (CESs) framework depends on the availability of efficient composition operators. For the analysis of modularly structured models, in general the subsystems have to be composed in the sense that a new single system is generated which represents the behavior of the interconnected modules. In the case of CESs, two operators for this purpose were introduced in the original work [1]: *cascade* and *feedback* interconnection. The idea was to successively apply these two operators to pairs of subsystems and finally obtain one composed system. When applied to examples, we discovered that this is not a convenient way to compose a complete block diagram. Moreover, when we tried to translate (timed) CESs into (timed) automata [2], we were looking for composition operators allowing a compositional translation. This means, first interconnected CESs are mapped block by block into equivalent automata, and then the connection is mapped into an equivalent synchronization rule. For these reasons, a new composition operator called *parallel interconnection* was developed which has several useful properties. It is a generalization of cascade and feedback and it can be mapped into a parallel product of automata. Furthermore, it is commutative, associative and monotonic (w.r.t. language inclusion).

For real-life systems, composition often is not feasible. In such cases one wants to be able to show properties of interconnected systems without building the composed system. A typical example of such a property is the refinement relation to some given specification. Based on the parallel interconnection operator and its monotonicity property, we defined a modular refinement rule which enables us to show that the parallel interconnection of two discrete CESs is a correct refinement of a given interconnection of two discrete CESs. This rule is in its spirit similar to the rules presented in e.g. [3, 4, 5].

The paper is organized as follows. In the next section, we recall basic definitions of CESs. In Section 3 we first revisit the cascade and feedback operators by Sreenivas and Krogh. Then, the parallel interconnection is introduced and it is shown how it generalizes the original operators. The modular refinement rule is presented in Section 4. In the conclusion, we summarize our contribution and discuss directions of future research.

2 Condition/event systems

Condition/event systems are introduced in [1] as a modeling paradigm for discrete event systems (DESs) and have gained increasing interest from the DESs and hybrid systems community (cf. [6], [7], [8]). The main motivation for developing CESs is to provide a framework for modeling DESs based on block diagrams and signal flows as is standard practice in system theory. A CES allows to model a system as a relation between inputs and output signals.

Given a nonempty finite set U, called *condition alphabet*, of *condition symbols*, a function $s_u : \mathbb{R}_{\geq 0} \to U$ is called *condition signal* over U, if it is right-continuous and finite-variable. A function f is called *finite-variable*, if f has a finite number of discontinuity points in every bounded subinterval of dom(f). The set of all condition signals over U is denoted by $C(U)$. Given a finite set V, called *event alphabet*, of *event symbols* containing a special *null symbol* $\mathbf{0}_V$, a function $s_v : \mathbb{R}_{\geq 0} \to V$ is called *event signal* over V, if $s_v(0) = \mathbf{0}_V$ and if there are at most finitely many points $t \in I$ such that $s_v(t) \neq \mathbf{0}_V$, for every bounded interval I in $\mathbb{R}_{\geq 0}$. The set of all event signals over V is denoted by $E(V)$. We call an event e with $e \neq \mathbf{0}_V$ *proper* event. Thus, condition signals are piecewise constant functions which associate with each time point a value in U; while event signals indicate the occur-

Proc. of the Fourth Workshop on Discrete Event Systems

rence of proper events only at single discrete points on the continuous time axis. Henceforth, we write $\mathbf{0}$ instead of $\mathbf{0}_V$, if the alphabet V is understood from the context. Moreover, if V is the product of alphabets $V_1, \cdots, V_n$, then we identify $\mathbf{0}_V$ with $(\mathbf{0}_{V_1}, \cdots, \mathbf{0}_{V_n})$. In the sequel, we use the following notation: Given a set X, we denote by $2^X_\emptyset$ the set of non-empty subsets of X.

Definition 1 *A* condition/event system *(CES) $\mathcal{S}$ is given by a tuple (U, V, Y, Z, S), where*

- *U is the* input condition alphabet*,*
- *Y is the* output condition alphabet*,*
- *V is the* input event alphabet*,*
- *Z is the* output event alphabet*, and*
- *$S : C(U) \times E(V) \to 2^{C(Y)\times E(Z)}_\emptyset$ is the* system behavior function.

We identify the function S with a *behavior relation* $S_R \subseteq \mathcal{B}(U, V, Y, Z) = C(U) \times E(V) \times C(Y) \times E(Z)$. An element of $\mathcal{B}(U, V, Y, Z)$ is called *behavior*. Henceforth, given a behavior $s \in \mathcal{B}(U, V, Y, Z)$, we denote by s_u, s_v, s_y, and s_z the signals such that $s = (s_u, s_v, s_y, s_z)$.

2.1 Discrete condition/event systems

CESs with discrete state realizations are CESs that can be described operationally.

Definition 2 *A* discrete CES *(DCES) is given by a tuple $(U, V, \mathcal{Q}, Y, Z, f, g, h, q_0)$, where*

- *U, V, Y and Z are as above,*
- *$\mathcal{Q}$ is a finite set of* states*,*
- *$f : \mathcal{Q}\times U \times V \to 2^{\mathcal{Q}}_\emptyset$ is the* state transition function*,*
- *$g : \mathcal{Q} \times U \to Y$ is the* condition output function*,*
- *$h : \mathcal{Q} \times \mathcal{Q} \times V \to Z$ is the* event output function*, and*
- *$q_0 \in \mathcal{Q}$ is the* initial state.

An event input signal can be used to force state transitions in a DCES in the sense that a state transition has to take place simultaneously with a synchronizing event indicated by the event input signal. Whereas, a condition input signal can be used to enable/disable transitions.

Definition 3 *A DCES is called* well-behaved*, if it satisfies the following conditions:*

1. Stuttering*: $\forall q \in \mathcal{Q}.\ \forall u \in U.\ q \in f(q, u, \mathbf{0})$, i.e., the system is allowed to stay in the same state as long as the value of the event input signal is the null symbol.*
2. Output triggering*: $\forall q \in \mathcal{Q}.\ h(q, q, \mathbf{0}) = \mathbf{0}$, i.e., the system can only output a proper event while remaining in the same state when it receives a proper input event.*

Henceforth, we only consider well-behaved DCESs and call them simply DCESs. To assign a semantics to DCESs, we introduce the notion of a *run*. For a finite-variable function x mapping non-negative reals to values in a set let $x(t^-)$ denote the limit from the left of x at t, i.e. $x(t^-) = \lim_{\epsilon\to 0} x(t - \epsilon)$.

Definition 4 *Given a DCES $D = (U, V, \mathcal{Q}, Y, Z, f, g, h, q_0)$, we call a* run *of D over $s \in \mathcal{B}(U, V, Y, Z)$ any right-continuous and finite-variable function $r : \mathbb{R}_{\geq 0} \to \mathcal{Q}$ which satisfies the following condition:*
for each $t \in \mathbb{R}_{\geq 0}$:

- *$r(t) \in f(r(t^-), s_u(t^-), s_v(t))$,*
- *$s_y(t) = g(r(t), s_u(t))$,*
- *$s_z(t) = h(r(t^-), r(t), s_v(t))$, and*
- *$r(0) = q_0$.*

The behavior relation S_D of a DCES D is the set of all behaviors $s \in \mathcal{B}(U, V, Y, Z)$ such that there exists a run of D over s. We write $\mathcal{S}_D$ to denote the CES (U, V, Y, Z, S_D).

3 Composition of condition/event systems

When a complex system is modeled using CESs, it is often divided into *modules*. Interaction of these modules is achieved by connections, these connections are condition and event signals. Moreover, when dealing with a network of interconnected CESs, one is interested in the semantics of this whole network. There are mainly two ways of defining this. One way is to define a *global* semantics which takes all single systems and a description of their interconnections and computes the network semantics following a given rule. In this case, the network is composed in a single step, and hence, reflects a flat structure. The other way is to define composition operators on CESs which take some of the systems and descriptions of how they are connected and replace these systems by a single one having their "overall" behavior. The result is a network of systems equivalent to the original one. Repeated application of these operators finally results in one CES having the overall semantics of the original network. Thus, a network described using such operators is composed in several steps to a single system and reflects a certain composition structure. In [1] two composition operators are defined, namely the *cascade interconnection* and the *feedback connection*. In the following, we introduce and discuss slight variants of these operators. We also introduce an operator, called *parallel operator*, which is more general than feedback and cascade.

3.1 Cascade interconnection

Given two CESs, the cascade interconnection describes the connection of the first system's outputs with the second system's inputs. The result is a system having the first system's inputs and the second system's outputs, and as semantics the functional concatenation of their behavior functions. In contrast to [1], in the resulting system the "intermediate" signals between both systems remain visible (cf. Figure 1).

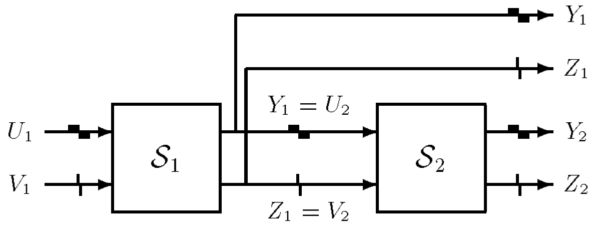

Figure 1: Cascade Interconnection of $\mathcal{S}_1$ and $\mathcal{S}_2$

Given a Cartesian product $B = B_1 \times \cdots \times B_n$ of sets and an element $b \in B$, we denote by b_i the projection of b on the ith component. Given two CESs $\mathcal{S}_i = (U_i, V_i, Y_i, Z_i, S_i)$, $i = 1, 2$, we say that $\mathcal{S}_1$ and $\mathcal{S}_2$ are *compatible*, if $Y_1 = U_2$ and $Z_1 = V_2$.

Definition 5 *Let $\mathcal{S}_i = (U_i, V_i, Y_i, Z_i, S_i)$, $i = 1, 2$, be two compatible CESs. A CES $\mathcal{S} = (U, V, Y, Z, S)$ is called* cascade interconnection *of $\mathcal{S}_1$ and $\mathcal{S}_2$, if and only if the following conditions hold:*

1. *$U = U_1$, $V = V_1$, $Y = Y_1 \times Y_2$, $Z = Z_1 \times Z_2$, and*
2. *for any $s \in \mathcal{B}(U, V, Y, Z)$, we have $s \in S$ iff $(s_{y_1}, s_{z_1}) \in S_1(s_u, s_v)$ and $(s_{y_2}, s_{z_2}) \in S_2(s_{y_1}, s_{z_1})$.*

Now, for two compatible DCESs we can always construct a DCES which is cascade interconnection of them (see [1]).

Theorem 1 *[1] There is an effective function that associates with every pair (D_1, D_2) of compatible DCESs a DCES D such that $\mathcal{S}_D$ is a cascade interconnection of $\mathcal{S}_{D_1}$ and $\mathcal{S}_{D_2}$.*

3.2 Feedback connection

Given a CES where the input and output alphabets are equal, for both condition and event signals, the feedback connection describes the connection of the system's outputs with the system's inputs ("feedback loop"). The signals in this loop remain visible. In contrast to [1], we have additional input and output signals in the system (cf. Figure 2). This makes the condition under which the feedback connection of a CES is well-defined slightly more complicated.

Definition 6 *Let $\mathcal{S} = (U_1 \times U_2, V_1 \times V_2, Y_1 \times Y_2, Z_1 \times Z_2, S)$ be a CES with $Y_2 = U_2$ and $Z_2 = V_2$. A CES $\mathcal{S}' = (U, V, Y, Z, S')$ is called* feedback connection *of $\mathcal{S}$, if and only if the following conditions hold:*

1. *$U = U_1$, $V = V_1$, $Y = Y_1 \times Y_2$, $Z = Z_1 \times Z_2$, and*
2. *for any $s \in \mathcal{B}(U, V, Y, Z)$, we have $s \in S'$ iff $\big((s_u, s_{y_2}), (s_v, s_{z_2}), s_y, s_z\big) \in S$.*

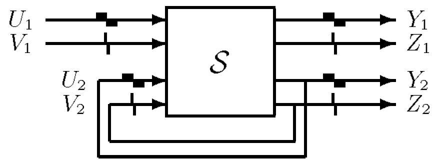

Figure 2: Feedback Connection of $\mathcal{S}$

In contrast to the cascade interconnection not every *discrete* CES can be transformed into a *discrete* CES that is a feedback connection of the original one. This is the case because there is not always a solution to the equations $g(q, u)_2 = u_2$ and $h(q, q', v)_2 = v_2$. Therefore, we have to require that there exists a solution to each of these equations. Here we require that there exists a *unique* solution. This leads to a smaller state space for the constructed DCES than in [1] where uniqueness is not required. However, as stated above, the system considered there does not have additional input signals, i.e., it only has the input signals U_2 and V_2. Allowing additional input signals leads to a more general feedback operator but makes the condition to check more complicated, since it then includes a quantification over all possible values of the input signals U_1 and V_1.

Definition 7 *Given a DCES $D = (U_1 \times U_2, V_1 \times V_2, \mathcal{Q}, Y_1 \times Y_2, Z_1 \times Z_2, f, g, h, q_0)$, with $Y_2 = U_2$ and $Z_2 = V_2$. We say that D satisfies the* unique solution property, *if the following conditions are satisfied:*

1. *For all $q \in \mathcal{Q}$ and $u_1 \in U_1$ there exists a unique $y_2 \in Y_2$ such that $g\big(q, (u_1, y_2)\big)_2 = y_2$.*
2. *For all $q, q' \in \mathcal{Q}$ and $v_1 \in V_1$ there exists a unique $z_2 \in Z_2$ such that $h\big(q, q', (v_1, z_2)\big)_2 = z_2$.*

Then, we have the following result:

Theorem 2 *Given a DCES D as in Def. 7 which satisfies the uniqueness solution property. A DCES D' can be effectively constructed such that $\mathcal{S}_{D'}$ is feedback connection of $\mathcal{S}_D$.*

3.3 Parallel interconnection

Though cascade and feedback allow to describe many systems, they are in general too restrictive. Therefore, we introduce an operator on CESs, called *parallel interconnection*, which generalizes both cascade and feedback.

To do so, we associate names with the input and output alphabets. Then, the parallel operator allows to interconnect systems by identifying signals which have the same name. The introduced parallel interconnection enjoys several interesting properties: it is commutative, associative, and monotonic with respect to language inclusion. Moreover, it allows to formulate a modular refinement principle for DCESs. This refinement principle is presented in Section 4.

In the following, consider n CESs $\mathcal{S}_1, \cdots, \mathcal{S}_n$, given by $\mathcal{S}_i = (\bar{U}_i, \bar{V}_i, \bar{Y}_i, \bar{Z}_i, S_i)$ with inputs $\bar{U}_i = \prod_{j=1}^{n_U^i} U_j^i$, $\bar{V}_i = \prod_{j=1}^{n_V^i} V_j^i$, and outputs $\bar{Y}_i = \prod_{j=1}^{n_Y^i} Y_j^i$, $\bar{Z}_i = \prod_{j=1}^{n_Z^i} Z_j^i$. We call U_j^i *alphabet* of $\mathcal{S}_i$ (similarly for V, Y, and Z). We assume that we have a unique name $\mathrm{N}(\beta)$ associated with every alphabet β of $\mathcal{S}_i$, i.e. within one single system, all associated names must be different. Alphabets of different systems are considered to be connected if they have the same name. In this case, they must contain exactly the same symbols. Moreover, all output alphabets must have different names, since outputs must not be connected with each other.

The parallel interconnection of CESs can now be defined as follows.

Definition 8 *A CES $\mathcal{S}$ with behavior S is called* parallel interconnection *of $\mathcal{S}_1, \cdots, \mathcal{S}_n$ if and only if the following conditions hold:*

1. *An alphabet is an input alphabet of $\mathcal{S}$ iff it is an input alphabet of some $\mathcal{S}_i$ but not the output alphabet of any $\mathcal{S}_j$. Moreover, every output alphabet of every $\mathcal{S}_i$ is an output alphabet of $\mathcal{S}$.*
2. *We have $s \in S$ if and only if there exist $s_1 \in S_1, \cdots, s_n \in S_n$ such that for every alphabet β of every S_i, the projections of s and s_i on β are equal.*

Now, we investigate the question whether the parallel interconnection of *discrete* CESs can be represented as a *discrete* CES, i.e. whether the class of DCESs is closed under parallel interconnection. We will show that this is the case under a certain *unique solution property* (which is an extension of the discrete feedback connection's property).

To formulate this property, we consider that the CESs $\mathcal{S}_1, \cdots, \mathcal{S}_n$ are given as discrete realizations D_i using states in $\mathcal{Q}_i$, transition functions f_i, and output functions g_i and h_i. Informally, the unique solution property states that for all states $q_1 \in \mathcal{Q}_1, \cdots, q_n \in \mathcal{Q}_n$ and for all possible values of the input alphabets of $\mathcal{S}$ (the parallel interconnection of $\mathcal{S}_1, \cdots, \mathcal{S}_n$), there exist *unique* values that the output functions g_i and h_i can take such that the equations for g_i and h_i are satisfied. For the sake of brevity, we give the formal definition of the unique solution property only for two DCESs D_1 and D_2 with connections from D_1 to D_2 and vice versa, and additional, independent inputs and outputs for each system (cf. Figure 3).

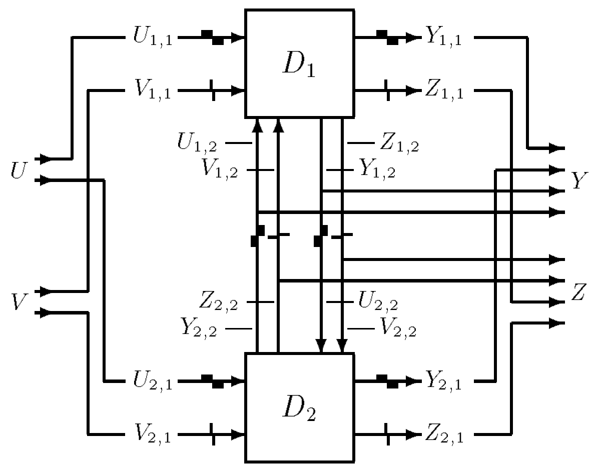

Figure 3: Parallel Interconnection of D_1 and D_2

Let D_i be given by $(U_i, V_i, \mathcal{Q}_i, Y_i, Z_i, f_i, g_i, h_i, q_{0,i})$ with inputs $U_i = U_{i,1} \times U_{i,2}$, $V_i = V_{i,1} \times V_{i,2}$, outputs $Y_i = Y_{i,1} \times Y_{i,2}$, $Z_i = Z_{i,1} \times Z_{i,2}$, for $i = 1, 2$, and $U_{1,2} = Y_{2,2}$, $V_{1,2} = Z_{2,2}$, $U_{2,2} = Y_{1,2}$, $V_{2,2} = Z_{1,2}$. Then, the *unique solution property* is satisfied for the parallel interconnection of D_1 and D_2 iff the following conditions are satisfied:

1. For all $q_1 \in \mathcal{Q}_1$, $q_2 \in \mathcal{Q}_2$, $u_{1,1} \in U_{1,1}$, $u_{2,1} \in U_{2,1}$, there exist unique $u_{1,2} \in U_{1,2}$ and $u_{2,2} \in U_{2,2}$, such that $g_1\big(q_1, (u_{1,1}, u_{1,2})\big)_2 = u_{2,2}$ and $g_2\big(q_2, (u_{2,1}, u_{2,2})\big)_2 = u_{1,2}$.
2. For all $q_1, q_1' \in \mathcal{Q}_1$, $q_2, q_2' \in \mathcal{Q}_2$, $v_{1,1} \in V_{1,1}$, $v_{2,1} \in V_{2,1}$, there exist unique $v_{1,2} \in V_{1,2}$ and $v_{2,2} \in V_{2,2}$, such that $h_1\big(q_1, q_1', (v_{1,1}, v_{1,2})\big)_2 = v_{2,2}$ and $h_2\big(q_2, q_2', (v_{2,1}, v_{2,2})\big)_2 = v_{1,2}$.

Then, we have the following theorem:

Theorem 3 *There is an effective function that associates with every DCESs $D_1, \cdots, D_n$ that satisfy the unique solution property a DCES D such that $\mathcal{S}_D$ is a parallel interconnection of $\mathcal{S}_{D_1}, \cdots, \mathcal{S}_{D_n}$.*

Using a hiding operator that hides output signals, we can show that the cascade and feedback operators can be expressed as parallel interconnections. The hiding operator is easily defined by projecting the output functions on the remaining signals. Then, we have the following results:

Lemma 1 *The feedback connection of DCESs is expressible in terms of parallel interconnection and hiding. The cascade interconnection of DCESs is expressible in terms of parallel interconnection.*

4 Modular refinement of condition/event systems

In this section, we present a modular rule that allows to show that a parallel interconnection $D_1 \parallel \cdots \parallel D_n$ of DCESs refines a parallel interconnection $D'_1 \parallel \cdots \parallel D'_n$. We say that D *refines* D', denoted $D \sqsubseteq D'$, if every behavior of D is also a behavior of D'. For the sake of brevity, we concentrate on the case where $n = 2$.

The straightforward way to prove

$$D_1 \parallel D_2 \sqsubseteq D'_1 \parallel D'_2 \quad (\dagger)$$

consists on computing a DCES D (resp. D') that is equivalent to $D_1 \parallel D_2$ (resp. $D'_1 \parallel D'_2$) for instance using the construction of Theorem 3 and then checking $D \sqsubseteq D'$ which can be done algorithmically. This strategy will, however, fail for real-life examples, since the system D is likely to be too large. Therefore, we present a modular refinement rule which is similar to the rules presented in [3, 9, 4]. The main idea of the rule is as follows. In order to prove (†), it suffices to prove

$$D_i \sqsubseteq D'_i$$

for $i = 1, 2$. This holds because the parallel interconnection operator is monotonic with respect to language inclusion. However, it is rarely the case that D_i refines D'_i in any environment. Usually, this will be the case for the environment consisting of the system D_j, for $j \neq i$, but not true in general. Therefore, the modular rule proposes to prove

$$D_1 \parallel D'_2 \sqsubseteq D'_1 \parallel D'_2 \text{ and } D_2 \parallel D'_1 \sqsubseteq D'_1 \parallel D'_2$$

in order to prove (†). Indeed, since D'_i is the abstract specification it has in general less states than D_i, and hence, the DCES corresponding to $D_1 \parallel D'_2$ has in general less states than $D_1 \parallel D_2$. This rule is, however, in general not sound as shown by the following example.

Example 1 *We consider the DCES D_1 given in Figure 4 with input event alphabet $V_{1,1} \times V_{1,2} = \{e\} \times \{a, c\}$, output event alphabet $Z_1 = \{b, d\}$, and input and output condition alphabets $U_1 = Y_1 = \{0\}$[1]. The set of states is $Q_1 = \{0, 1\}$. For all states and all input values not*

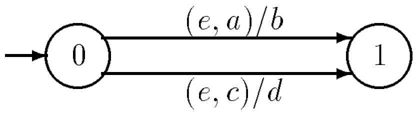

Figure 4: D_1

explicitly given in the figure, we assume a self-loop transition with output **0**.

Moreover, we define a DCES D_2 given in Figure 5 with input event alphabet $V_2 = \{b, d\}$, output event alphabet $Z_2 = \{a, c\}$, input and output condition alphabets $U_2 = Y_2 = \{0\}$, and set of states $Q_2 = \{0, 1\}$.

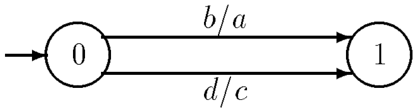

Figure 5: D_2

Furthermore, we define two DCESs D'_i, $i = 1, 2$, given in Figure 6.

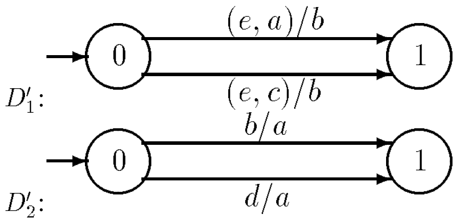

Figure 6: D'_1 and D'_2

The DCES D'_1 (resp. D'_2) has the same alphabets as D_1 (resp. D_2). Suppose we want to prove $D_1 \parallel D_2 \sqsubseteq D'_1 \parallel D'_2$, where D_1 and D_2 are put in parallel by connecting $V_{1,2}$ with Z_2 and V_2 with Z_1; and similarly for D'_1 and D'_2. Then, we have $D_1 \parallel D'_2 \sqsubseteq D'_1 \parallel D'_2$ and $D'_1 \parallel D_2 \sqsubseteq D'_1 \parallel D'_2$ but not $D_1 \parallel D_2 \sqsubseteq D'_1 \parallel D'_2$, since $D_1 \parallel D_2$ may generate the output event c while this is not possible for the system $D'_1 \parallel D'_2$.

Thus, we need a condition to break the circular reasoning of this rule. To define such a condition, we introduce the following definition. We say that D_i *depends* on D_j, for $i \neq j$, denoted by $D_j < D_i$, if some output event signal of D_j is an input of D_i. Now, the condition that makes the modular rule above sound is that the dependency relation $<$ is asymmetric. If we have n systems interconnected in parallel then the condition is that the transitive closure $<^+$ of $<$ is irreflexive, i.e. for no system D_i, we have $D_i <^+ D_i$. To summarize we have the following result:

Lemma 2 *If $<^+$ is irreflexive, $D_1 \parallel D'_2 \sqsubseteq D'_1 \parallel D'_2$, and $D_2 \parallel D'_1 \sqsubseteq D'_1 \parallel D'_2$ then $D_1 \parallel D_2 \sqsubseteq D'_1 \parallel D'_2$.*

[1] These sets are singletons. That means there is only one condition which thoroughly holds. Therefore they are omitted in the graphical description.

5 Conclusion

We introduced a parallel operator for CESs which generalizes the cascade and feedback operators of [1]. We showed that under assumption of the unique solution property the class of DCESs is closed under parallel interconnection. It can be shown that this closure property also holds when uniqueness is not required. The parallel operator enjoys the nice properties of commutativity, associativity, and monotonicity.

Moreover, we presented a modular refinement rule that allows to show that the parallel interconnection of DCESs is a correct implementation/refinement of a given parallel interconnection of DCESs. We plan to extend the parallel operator and its refinement rule to timed CESs.

Acknowledgment

This work has been supported by the German Research Council (DFG) within the special program KONDISK (Analysis and Synthesis of Technical Systems with Continuous-discrete Dynamics) under the grants Ro 1122/2 and En 152/19.
URL: http://www.informatik.uni-kiel.de/˜kondisk/

References

[1] R.S. Sreenivas and B.H. Krogh. On Condition/Event Systems with Discrete State Realizations. In *Discrete Event Dynamic Systems: Theory and Applications 1*, pages 209–236. Kluwer Academic, Boston, USA, 1991.

[2] R. Huuck, Y. Lakhnech, L. Urbina, S. Engell, S. Kowalewski, and J. Preußig. Comparing Timed Condition/Event Systems and Timed Automata. In *HART'97*, LNCS 1201, pages 81–86. Springer–Verlag, 1997.

[3] M. Abadi and L. Lamport. Composing Specifications. *ACM Trans. on Prog. Lang. and Sys.*, 15(1):73–132, 1993.

[4] R. Alur and T. Henzinger. Reactive Modules. In *11th IEEE Symposium on Logic in Computer Science*, pages 207–218, 1996.

[5] K.L. McMillan. A Composition Rule for Hardware Design Refinement. In O. Grumberg, editor, *CAV'97*, LNCS 1254, pages 24–35. Springer Verlag, 1997.

[6] H.-M. Hanisch, A. Lüder, and M. Rausch. Controller Synthesis for Net Condition/Event Systems with a Solution to Incomplete State Observation. *European Journal of Control*, 3:280–291, 1995.

[7] L.E. Holloway. Time Measures and State Maintainability for a Class of Composed Systems. In *WODES'96*, pages 24–30, Edinburgh, UK, 1996.

[8] S. Engell and I. Hoffmann. Modular Hierarchical Models of Hybrid Systems. In *35th IEEE Conf. on Decision and Control*, Kobe, Japan, 1996.

[9] R. Gawlick, R. Segala, J. Sogaard-Andersen, and N. Lynch. Liveness in Timed and Untimed Systems. In *ICALP*, volume 820 of *LNCS*. Springer Verlag, 1994.