Ensuring Properties of Interaction Systems

G. Gössler[1], S. Graf[2], M. Majster-Cederbaum[3]*, M. Martens[3], J. Sifakis[2]

[1] INRIA Rhône-Alpes
Montbonnot, France
gregor.goessler@inria.fr

[2] VERIMAG
Grenoble, France
{graf,sifakis}@imag.fr

[3] University of Mannheim
Mannheim, Germany
mcb@informatik.uni-mannheim.de

Abstract. We propose results ensuring properties of a component-based system from properties of its interaction model and of its components. We consider here deadlock-freedom and local progress of subsystems. This is done in the framework of interaction systems, a model for component based modelling described in [9]. An interaction system is the superposition of two models: a behavior model and an interaction model. The behavior model describes the behavior of individual components. The interaction model describes the way the components may interact by introducing connectors that relate actions from different components. We illustrate our concepts and results with examples.

1 Introduction

Component-based design techniques are important for mastering design complexity. Nevertheless, for these techniques to be useful, it is essential that they guarantee more than syntax-based interface compatibilities. Methods based on the assume-guarantee paradigm [15] or similarly on the more recent interface automata [4] are useful for the verification of safety properties provided that they can be easily decomposed into a conjunction of component properties.

We show how one can discuss properties such as (global) deadlock-freedom and progress of a subset of components in a framework for component-based modelling by making use of compositional methods in various ways. Given that violations of safety properties can be expressed as deadlocks, these results can be also applied for general safety properties.

In previous papers [8,9,7,16], a framework for component-based modelling was proposed which clearly separates interaction from behavior. An **interaction model** describes how system components can interact. A **behavior model** is used to describe the behavior of individual components. The aim of this framework is twofold. One is to allow compositional verification. The second aim is to provide a composition framework with a flexible means for controlling the collaboration of a set of components. A general framework for defining such glue operators was presented in [16] and its main ingredients are the interaction model presented here and priority rules, which are not considered in this paper.

Here, we generalize the initial results of [9] for proving deadlock freedom and local progress, 1) to apply to a broader class of systems and 2) to apply to subsystems. In addition, we adapt the framework to support bottom-up system development. Hence, we may start with some interaction systems that exhibit certain desirable properties.

* while working on this paper the author was a guest at and supported by the Ecole Polytechnique in Palaiseau

These can be combined to build more complex systems. We may now ask under which conditions the desirable properties can be ensured for the composed system.

We present and illustrate here the central notions and results concerning deadlock-freedom and progress on a simple version of the framework without variables.

2 Connectors, Interaction Models and Interaction Systems

We consider a framework where components i in a set K of components together with their port sets $\{A_i\}_{i\in K}$ are the basic building blocks. Components can interact, that is cooperate. A set C of connectors controls the cooperation. A connector is a set of ports with at most one port of each component, and an interaction is a subset of a connector. As an example, we consider a system with three components 1, 2, 3 and an interaction $\alpha = \{a, b, c\}$, where a is a port of component 1, b a port of component 2 and c a port of component 3. The interaction α describes a step of the system where a, b, and c are performed simultaneously. Each component i may constrain the order in which interactions on its ports can take place. We consider here these constraints to be given in the form of a transition system with edges labelled by elements in the port set.

Definition 1
A component system $CS = (K, \{A_i\}_{i\in K})$ *consists of a set K of* components *and has for each component $i \in K$ a* port set A_i*, that is disjoint from the port set of every other component. Ports are also referred to as* actions.
The union $A = \bigcup_{i\in K} A_i$ of all port sets is the port set of K. A finite nonempty subset c of A is called a connector *for CS, if it contains at most one port of each component $i \in K$. A* connector set *is a set C of connectors for CS that covers all ports, and where no connector contains any other:*

a) $\bigcup_{c\in C} c = A$
b) $c \subseteq c' \Rightarrow c = c' \quad$ *for all* $c, c' \in C$.

If c is a connector, $I(c)$ denotes the set of all nonempty subsets of c and is called the set of interactions of c. *For a set C of connectors,*

$$I(C) = \bigcup_{c\in C} I(c)$$

is the set of interactions of C. If C is a connector set, it is clear by the above that the connectors $c \in C$ are the maximal elements in $I(C)$. For component i and interaction α, we put $i(\alpha) = A_i \cap \alpha$. We say that component i participates *in α, if $i(\alpha) \neq \emptyset$.*

Remark 1
A connector $c = \{a\}$, $a \in A_i$, consisting of a single action, can be identified with this action. It models the situation that a is considered as internal action of component i that takes place independently of the environment.

In the following, we always assume that $K = \{1, ..., n\}$ for some $n \in \mathbb{N}$ or that K is countably infinite.

Example 1
We consider three components $1, 2, 3$ *with port sets* $A_1 = \{a_1, a_2, a_3\}$, $A_2 = \{b_1, b_2, b_3\}$, *and* $A_3 = \{c_1, c_2, c_3\}$. *The connector set*

$$C = \{\{a_1, b_1\}, \{b_1, c_1\}, \{a_1, c_1\}, \{a_2, b_2, c_2\}, \{a_3\}, \{b_3\}, \{b_2, c_3\}\}$$

describes a situation where any two systems may cooperate via their first port or they cooperate all via their second port. Components 1 *and* 2 *may act individually via their third port. Finally component* 2 *may cooperate with the third port of component* 3 *via its second port. This situation can be graphically displayed by Figure 1 where a connector* c *with* $|c| > 1$ *is represented by a line connecting its ports.*

Remark 2
Please note, that connectors allow a very liberal form of cooperation. One action may cooperate with m_1 *other actions in one connector whereas it cooperates with* m_2 *actions in a different connector. In the above example this is the case for action* b_2.

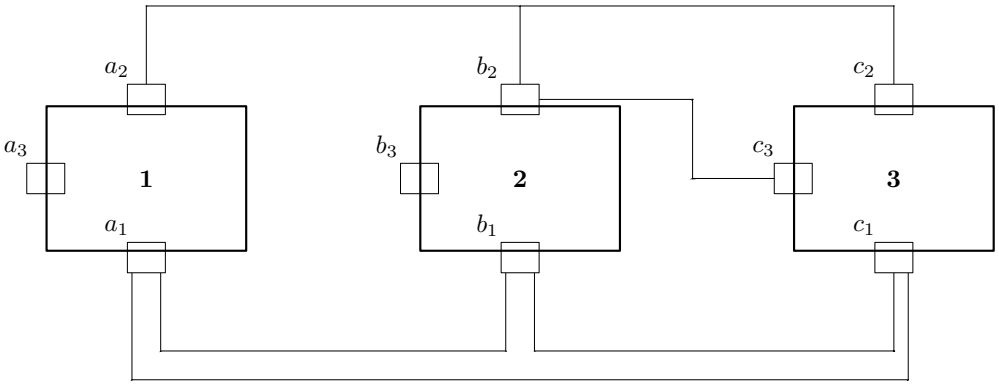

Fig. 1. Example of connectors

When we have specified for a component system (by choosing a connector set C) how the components can interact, we want to state which interactions should be considered independent of the availability of actions of other components.

In the example above, one design decision could be to declare the interactions $\{a_2\}$ and $\{b_1\}$ independent. That is, no matter if the actions occurring in a connector involving one of these actions are available or not, a_2 respectively b_1 may be performed independently of the environment, i. e. the status of other components. For this purpose, we introduce the notion of *complete* interactions and *interaction model*.

An *interaction model* for a component system CS is defined by a connector set C together with a set $Comp$ of interactions that are declared to be *complete*. If an

interaction is declared *complete*, it can be performed independently of the environment. By environment we mean the other components and potential extensions of the system. In [9] it is required that all supersets of a complete interaction in $I(C)$ should also be complete[1], that is, $Comp$ has to be closed with respect to $I(C)$ in the following sense.

Definition 2
Let U, T be sets of sets, $U \subseteq T$. Then U is closed w.r.t. *T, if for any $u \in U$ it contains all supersets $t \in T$ of u. The* closure *of U w.r.t. T, $cl(U,T)$, is the smallest set that contains U and is closed w.r.t. T.*

Definition 3
Let C be a connector set for the component system CS. If $Comp \subseteq I(C)$ is closed with respect to $I(C)$, then

$$IM = (C, Comp)$$

is called an interaction model for *CS. The elements of $Comp$ are called* complete interactions.

Example 1 continued:
By choosing

$$Comp = cl(\{\{a_2\}, \{b_1\}\}, I(C)) =$$

$$\{\{a_2\}, \{a_2, b_2\}, \{a_2, c_2\}, \{a_2, b_2, c_2\}, \{b_1\}, \{a_1, b_1\}, \{b_1, c_1\}\}$$

we model the situation described above.

As we stated before, we assume in this paper that the local behavior of each component $i \in K$ of a component system is given by a transition system T_i. When the connector set C is fixed, the *global behavior* of the system is given by allowing in each global state those transitions that correspond to interactions in $I(C)$.

Definition 4
Let $CS = (K, \{A_i\}_{i \in K})$ be a component system and $IM = (C, Comp)$ an interaction model for CS.
Let for each component $i \in K$ a transition system $T_i = (Q_i, A_i, \rightarrow_i)$ be given, where $\rightarrow_i \subseteq Q_i \times A_i \times Q_i$. We write $q \xrightarrow{a}_i q'$ for $(q, a, q') \in \rightarrow_i$. We suppose that $Q_i \cap Q_j = \emptyset$ for $i \neq j$.
The induced interaction system *is given by*

$$Sys = (CS, IM, T),$$

where the global behavior $T = (Q, I(C), \rightarrow)$ is obtained from the behaviors of individual components, given by transition systems T_i, in a straightforward manner:

- *$Q = \prod_{i \in K} Q_i$, the cartesian product of the Q_i, which we consider to be order independent. We denote states by tuples $(q_1, ..., q_j, ...)$ and call them* global states.
- *the relation $\rightarrow \subseteq Q \times I(C) \times Q$, defined by*
  $\forall \alpha \in I(C)\ \forall q, q' \in Q : [\ q = (q_1, ..., q_j, ...) \xrightarrow{\alpha} q' = (q'_1, ..., q'_j, ...)$ iff
  $\forall i \in K\ \ (q_i \xrightarrow{i(\alpha)}_i q'_i$ if i participates in α and $q'_i = q_i$ otherwise)].

[1] Please note, that most results carry over to a situation where we drop this condition. The results in Section 5 have to be slightly modified if we work in this more general setting.

A state $q_i \in Q_i$, resp. a global state $q \in Q$, is called complete, *if there is some interaction $\alpha \in C \cup Comp$ and some q_i' with $q_i \xrightarrow{\alpha}_i q_i'$, resp. some q' with $q \xrightarrow{\alpha} q'$. Otherwise it is called* incomplete.

Note that a global state $q = (q_1, q_2, ...)$ is complete if q_i is complete for some i. But q may be complete even if all q_i are incomplete.

Please also note that we allow edges to be labelled by elements that are neither maximal nor complete in the definition of T. For Sys itself we will only be interested in transitions labelled with $\alpha \in C \cup Comp$ as those are independent of the environment. When, however, we compose interaction systems as described in Section we will need the information about the transitions labelled with elements in $I(C)$.

Remark 3
A connector $c = \{a_1, ..., a_l\}$ specifies a degree of cooperation. For this connector to be performed in the global system, all l partners have to cooperate. As different connectors may have different size and involve different components, the degree of cooperation and the involved partners vary in the system. For instance, in one global state m_1 components may cooperate via one connector and alternatively m_2 components may cooperate via some other connector. In another state yet another type of cooperation is possible. Also, one port may cooperate in different connectors with different partners and different degrees of cooperation. Note that this is a very interesting feature of the model which allows for great flexibility and distinguishes our framework from others, for example process algebras or I/O-automata [10]. In process calculi such flexibility is either not realizable or can be achieved only in a clumsy way.

Example 2
Consider a set of components $Worker_i$, $1 \leq i \leq n$, that may choose between the execution of two tasks, $t0_i$ and $t1_i$. Each component $Worker_i$ can do its task $t0_i$ independently of the others, but has to cooperate with component $Control_1$ counting the number of tasks $t0$ already started. For executing the task $t1_i$, the component $Worker_i$ needs the collaboration of $Control_{21}$ or of $Control_{22}$ for the whole duration of the execution of $t1_i$. As the execution of a task may have some duration such that during its execution other interactions may take place, each task execution is represented by a corresponding start and end event. The definition of the components together with the local transition systems is provided in Figure 2.

In order to achieve the collaboration of these components, we consider the interaction model $IM_1 = (C, Comp)$ with the connector set

$$C = \{conn_{1i}, conn_{2i}, conn_{3si}, conn_{3ei}, conn_{4si}, conn_{4ei} | 1 \leq i \leq n\}$$

and $Comp = \emptyset$. Here

$$\begin{array}{ll} conn_{1i}: & \{count, st\_t0_i\}, \text{ for all } i \in \{1...n\} \\ conn_{2i}: & \{e\_t0_i\}, \text{ for all } i \in \{1...n\} \\ conn_{3si}: & \{start1, st\_t1_i\}, \text{ for all } i \in \{1...n\} \\ conn_{3ei}: & \{end1, e\_t1_i\}, \text{ for all } i \in \{1...n\} \\ conn_{4si}: & \{start2, st\_t1_i\}, \text{ for all } i \in \{1...n\} \\ conn_{4ei}: & \{end2, e\_t1_i\}, \text{ for all } i \in \{1...n\} \end{array}$$

All connectors represent binary rendezvous or local actions, which can easily be expressed in process algebra. However, the actions representing the start and end of

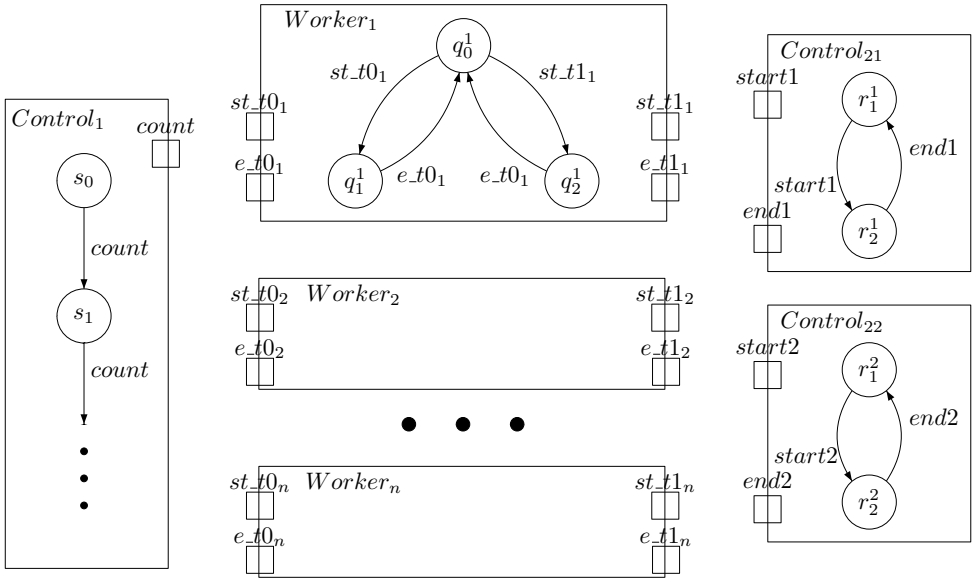

Fig. 2. Worker example: components with ports and component behavior

execution of task $t1_i$ *can synchronize either with* $Control_{21}$ *due to the connectors* $conn_3$ *or with* $Control_{22}$ *due to the connectors* $conn_4$*. This can not be directly expressed in CCS- or TCSP-style process algebra. Also, a third control unit* $Control_{23}$ *for improving the performance of the system, could be added without changing the behavior of the worker components. In this interaction system e.g. any global state* q *containing* q_1^i*, for some* i*, as well as* $(s_l, q_2^1, ..., q_2^n, r_1^1, r_2^2)$ *is complete for any* l*, whereas e.g. the states* $(s_l, q_2^1, ..., q_2^n, r_1^1, r_1^2)$ *are incomplete for any* l*. We can modify this system* $Sys = (CS, IM_1, T)$ *in various ways. One may modify the local behavior while maintaining the interaction model. Or one may conceive a different scheme for the interaction. For example, instead of interleaving the terminations of the tasks* $t0_i$*, we may also allow executing them in cooperation; this can be done by replacing the* n *connectors* $conn_{2i}$ *by a single connector* $conn_2$*, leading to interaction model* IM_2*:*

$$conn_2 : \{e\_t0_1, e\_t0_2, ..., e\_t0_n\}$$

and declare each individual action to be complete. In this modelling when any number of workers is ready to terminate they may do so simultaneously.

3 Properties of Interaction Systems

We consider in the following two essential properties of interaction systems and show in the next sections how they can be established by either testing the property using a graph criterion or by deriving the property from properties of subsystems. In what follows, we consider a system

$$Sys = (CS, IM, T) \text{ with}$$

$$CS = (K, \{A_i\}_{i \in K}) \text{ and } IM = (C, Comp) \text{ and } T = (Q, I(C), \rightarrow).$$

where T is constructed from given transition systems T_i, $i \in K$, as described in Definition 4.

The first property under consideration is (global) deadlock-freedom. A system is considered to be (globally) deadlock-free if in every global state it may perform a maximal or complete interaction, in other words, if every global state is complete. This definition is justified by the fact that both for complete and maximal interactions there is no need to wait for other components to participate. In the case of maximal interactions there do not exist such components, in the case of complete interactions this holds true by the definition of an interaction model. If a maximal or complete interaction is enabled in a global state q, it may be performed right-away. A global state q where neither a maximal or complete interaction may be performed means that every component needs some other components' cooperation which do not provide the needed ports in q.

Definition 5
An interaction system Sys is called deadlock-free *if for every state $q \in Q$ there is a transition $q \xrightarrow{\alpha} q'$ with $\alpha \in C \cup Comp$.*

In many systems there is a designated start-state q_0 and one is only interested in the states that can be reached from q_0. To model this and similar situations we introduce a notion of P-deadlock-freedom in [6], where P is a predicate on the state space and the existence of a transition labelled by some $\alpha \in C \cup Comp$ is only requested for states satisfying P. For $P = true$ we obtain the above notion of deadlock-freedom.

Deadlock-freedom is an important property of a system. But it does not provide any information about the progress that an individual component $i \in K$ may achieve. Hence, it is interesting to consider the property of (individual) progress of component i, i.e. the property that at any point of any run of the system, there is an option to proceed in such a way that i will eventually participate in some interaction, which means that a clever scheduler can achieve progress of component i.

Definition 6
Let Sys be a deadlock-free interaction system. A run of Sys is an infinite sequence σ

$$q_0 \xrightarrow{\alpha_0} q_1 \xrightarrow{\alpha_1} q_2 \ldots$$

with $q_l \in Q$ and $\alpha_l \in C \cup Comp$. For $n \in \mathbb{N}$, σ_n denotes the prefix

$$q_0 \xrightarrow{\alpha_0} q_1 \xrightarrow{\alpha_1} q_2 \ldots \xrightarrow{\alpha_{n-1}} q_n$$

We define here a notion of progress of subsets $K' \subseteq K$ of components in two ways. In the first case, we just guarantee that the system may always proceed in such a way that some component of K' participates in some interaction. In the second case, it may proceed in such a way that every component $i \in K'$ participates in some interaction.

Definition 7
Let Sys be a deadlock-free interaction system. Let $K' \subseteq K$.

- *K'* may progress in *Sys, if for any run σ of Sys and for any $n \in \mathbb{N}$ there exists σ' such that $\sigma_n \sigma'$ is a run of Sys and for some $i \in K'$, i participates in some interaction α of σ'.*

– K' may strongly progress in Sys, *if for any run* σ *of* Sys *and for any* $n \in \mathbb{N}$ *there exists* σ' *such that* $\sigma_n\sigma'$ *is a run of* Sys *such that every* $i \in K'$ *participates in some interaction* α *of* σ'.

If a set K' of components *may progress* in Sys then a clever scheduler can guarantee that a *run is chosen* where infinitely often some interaction with participation of the subsystem K' is performed.

If $|K'| = 1$ then the two notions coincide and yield the special case presented in [9]. As for deadlock-freedom one may generalize the progress properties to P-progress.

In the following example, we look at some of the properties defined above. For this example, we introduce the following rule of maximal progress.

Definition 8
The maximal progress rule *restricts the transition relation for* Sys *to maximal transitions, i.e. to those transitions such that* $q \xrightarrow{\alpha} q'$, *implies that there is no* β, q'' *with* $\alpha \subsetneq \beta$ *and* $q \xrightarrow{\beta} q''$.

Example 3
We consider a system of n *identical tasks that have to be scheduled, differently to the preceding example, by allowing preemption and without explicit representation of a* scheduler *or a* controller. *In our framework, we achieve this by collaboration of the* n *tasks with an appropriate interaction model.*

We consider a set of tasks i *(*$i \in K = \{1, ..., n\}$*) that compete for some resource in mutual exclusion. The transition system* T_i *of each task* i *is given in Figure 3 and needs not to be further explained. Let the set of ports of component* i *be:*

$$A_i = \{activate_i, start_i, resume_i, preempt_i, finish_i, reset_i\}$$

We want to guarantee mutual exclusion with respect to the exec state, i.e. no two tasks should be in this state at the same time, in the sense that this is an inductive invariant [2]. *Mutual exclusion, in this sense, can be achieved using the rule of maximal progress and the interaction model* $IM = (C, Comp)$ *with the connector set* $C = \{conn_1^i, conn_2^{ij}, conn_3^{ij}, conn_g\}$, *where*

$$\begin{aligned} conn_1^i&: \{activate_i\},\ i \in K \\ conn_2^{ij}&: \{preempt_i, start_j\},\ i, j \in K, i \neq j \\ conn_3^{ij}&: \{resume_i, finish_j\},\ i, j \in K, i \neq j \\ conn_g&: \{reset_1, ..., reset_n\} \end{aligned}$$

and $Comp = cl(\{\{start_j\}, \{finish_j\} | 1 \leq j \leq n\}, I(C))$.

Mutual exclusion is guaranteed because whenever component j *enters* $exec_j$, *either by* $start_j$ *or* $resume_j$, *then either there is no other task in its exec-state or the component* i *that is in the state* $exec_i$ *must leave this state. The following items explain why this is the case for each of the two transitions entering the exec-state:*

– *for* $resume_j$, *the reason is that* $resume_j$ *can never happen alone. It can only be executed together with the* $finish_i$ *action if component* i *is currently in the critical state* $exec_i$.

[2] whenever a global state satisfies this condition then any successor state should satisfy it as well

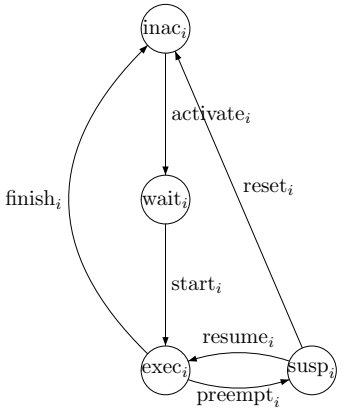

Fig. 3. Transition system of task i

– *for $start_j$, which is complete, the reason is the rule of maximal progress: when component i is in the critical state $exec_i$, it can execute the $preempt_i$ action. Therefore, $start_j$ cannot be executed alone as also the pair $\{preempt_i, start_j\}$ is enabled. On the other hand, if there is no process in the critical section, component j can enter it by executing $start_j$ alone.*

We now consider the properties of deadlock-freedom and progress. The interaction system $Sys = (CS, IM, T)$ as defined above, where we first ignore the rule of maximal progress, is deadlock-free as the only incomplete state in each local transitions system T_i is the state $susp_i$. But the global state in which all components are in state $susp_i$ admits the interaction $conn_g = \{reset_1, ..., reset_n\} \in C$ and hence does not cause any problems. Each component i may progress.

When a finite interaction system is deadlock-free, then it is also deadlock-free when we apply the rule of maximal progress. This is the case because, even if we disallow some transitions in a global state q, as they are not maximal there is always at least one transition with a label in $C \cup Comp$ left. Hence, our example is deadlock-free under the rule of maximal progress. Also every component may progress under the rule of maximal progress. As all components have identical behavior it suffices to consider one of them, say component 1. The only situation in which component 1 cannot proceed by itself is when it is in state $susp_1$. We have to show that we can reach a global state where it can perform a transition:

– *case 1) all other components are in the state susp. Then $conn_g$ can happen and component 1 has proceeded.*
– *case 2) at least one component j is in the state $exec_j$. Then $\{resume_1, finish_j\}$ may happen.*

– *case 3) not all other components are in state susp and none is in state exec. Then there must be one component j that is in state $inac_j$ or $wait_j$. If it is in state $inac_j$ then it performs the complete action $activate_j$ and reaches state $wait_j$. As there is no component in state exec there is no preempt action available and $start_j$ may be performed alone even under the rule of maximal progress. Now, $\{resume_1, finish_j\}$ may happen and component 1 has made progress.*

In this example, we enforced deadlock-freedom by introducing the connector $conn_g$, i.e., by solving the problem on the level of the interaction model. An alternative way is to consider an invariant on the state space, namely the one expressing that not all local states are *susp*-states. One then shows that if we start in a state that satisfies this invariant then all successor states satisfy the invariant too. Hence, the problematic state is never reached from a good state.

In [6] we introduce further properties of interaction systems, in particular local deadlock-freedom, liveness, fairness, and robustness of properties with respect to failure of ports/components.

4 Testing Deadlock-Freedom and Progress

The definition of deadlock-freedom and other properties of interaction systems are conditions on the global state space and hence cannot be tested directly in an efficient way. In [13] it was shown that deciding deadlock-freedom in interaction systems is NP-hard. In [12] it was shown that deciding liveness is NP-hard. Therefore it is desirable to establish (stronger) conditions that are easier to test and entail the desired properties. In [6] we present a condition that can be tested in polynomial time and ensures liveness of a set of components. In [11] we gave a parameterized condition that can be tested in polynomial time and ensures (local) deadlock-freedom of an interaction system. Here we present a generalization of a criterion developed in [9] that ensures deadlock-freedom and give a condition that guarantees local progress of a set of components.

In what follows, we assume for simplicity that the local transition systems T_i have the property that they offer *at least one action* in every state. The general case can be reduced to this case by introducing idle actions or by adapting the definitions and results below to include this situation.

In [9], a condition for deadlock-freedom of Sys (called interaction safety there) was presented that uses a directed graph with labels in $A = \bigcup A_i$. The set of nodes is given by

$$V = K \cup H$$

where K is the set of components and H is some subset of $C \times Comp$. The edges relate nodes in K with nodes in H and vice versa. The non-existence of certain cycles in the graph ensures deadlock-freedom of the system. We propose here another graph called G_{Sys} that is simpler and smaller. One can exhibit a system with n components where the graph of [9] contains $O(n^3)$ nodes and $\Omega(n^3)$ edges independently of the structure of the transition systems. In contrast to this G_{Sys} has n nodes and - depending on the local transition systems - possibly no edges. Based on G_{Sys} we establish a criterion involving a notion of refutability that

– allows classifying a larger set of systems as deadlock-free
– is suitable to give a characterization of all deadlock-free systems.

In the following we define the labelled directed graph G_{Sys}. The nodes of this graph are the components of Sys. There are two kinds of labels for the edges. An edge (i, c, j), where $c \in C$, means that there is an incomplete state $q_i \in Q_i$ such that $i(c)$ is enabled in q_i and j participates in c. Similarly an edge $(i, (c, \alpha), j)$ means that there is an incomplete state $q_i \in Q_i$ such that $i(c)$ is enabled in q_i and j participates in α. Both edges mean that it might happen in some global state $(\ldots, q_i, \ldots)$ that i has to wait for j.

Definition 9
Let Sys be an interaction system. The dependency graph *for Sys is a labelled directed graph*

$$G_{Sys} = (V, E)$$

where the set of nodes is $V = K$ and the set of labels is $L = L_1 \cup L_2$, with

$$L_1 = \{c \in C \mid \nexists \alpha \in Comp : \alpha \subseteq c\}$$

and

$$L_2 = \{(c, \alpha) \mid \ c \in C, \alpha \in Comp, \alpha \subseteq c \text{ and } \ \nexists \beta \in Comp : \beta \subsetneq \alpha\}$$

and the set of edges is $E \subseteq V \times L \times V$ such that

a) $(i, c, j) \in E$, *where* $c \in L_1$, *iff*
$\exists q_i \in Q_i,\ q_i$ *incomplete,* $\exists q_i' \in Q_i$ *such that* $q_i \overset{i(c)}{\rightarrow} q_i'$ *and* $j(c) \neq \emptyset$
b) $(i, (c, \alpha), j) \in E$, *where* $(c, \alpha) \in L_2$, *iff*
$\exists q_i \in Q_i,\ q_i$ *incomplete,* $\exists q_i' \in Q_i$ *such that* $q_i \overset{i(c)}{\rightarrow} q_i'$ *and* $j(\alpha) \neq \emptyset$.

In addition to G_{Sys}, resp. subgraphs G of G_{Sys}, we will refer to snapshots of G_{Sys}, resp. G, with respect to a global state $q = (q_1, q_2, \ldots, q_i, \ldots) \in Q$

$$G_{Sys}(q) = (V, E(q))$$

where $E(q) \subseteq E$ such that

a) $(i, c, j) \in E(q)$, *where* $c \in L_1$, *iff*
q_i *is incomplete and* $\exists q_i' \in Q_i$ *such that* $q_i \overset{i(c)}{\rightarrow} q_i'$
b) $(i, (c, \alpha), j) \in E(q)$, *where* $(c, \alpha) \in L_2$, *iff*
q_i *is incomplete and* $\exists q_i' \in Q_i$ *such that* $q_i \overset{i(c)}{\rightarrow} q_i'$

Moreover, we introduce the snapshot $G^a_{Sys}(q) = (V, E^a(q))$ relative to q and $a = (a_1, ..., a_i, ...)$ with $a_i \in A_i$. $G^a_{Sys}(q)$ contains those edges (i, c, j), resp. $(i, (c, \alpha), j)$, of $G_{Sys}(q)$, where $a_i = i(c)$ and we apply the same constructions to subgraphs of G_{Sys}.

Remark 4
Note that for the construction of the graph we inspect each local transition system T_i separately and hence avoid the combinatorial complexity of global state analysis. In the finite case, i.e. $K = n$, A_i finite and T_i finite for $i = 1, ..., n$, the graph G_{Sys} can be constructed in cost polynomial in $|C|$, $|Comp|$, and the sum of the sizes of the local transition systems.

In the following we will use predicates on global states that are conjunctions of predicates on local states.

Notation 1
Let for $i \in K$ $pred_i$ be a state predicate *on Q_i. We say $q = (q_1, q_2, \ldots) \in Q$ satisfies $pred_i$ if q_i satisfies $pred_i$.*

We use the following predicates on states. $en(a_i)$ describes all states of component i where action a_i is enabled. $cond(e)$ for edge $e = (i, c, j)$ describes all global states $q = (q_1, \ldots, q_i, \ldots)$ such that $i(c)$ is enabled in q_i and there is some action a_k in c that is not enabled in the respective local state of q. $inc(i)$ yields all incomplete states of component i.

Definition 10
For $a_i \in A_i, i \in K$: $en(a_i) = \{q_i \in Q_i \mid q_i \xrightarrow{a_i} q_i' \text{ for some } q_i'\}$.
For $e = (i, c, j) \in E$: $cond(e) = en(i(c)) \wedge (\exists x \in c : \neg en(x))$.
For $e = (i, (c, \alpha), j) \in E$: $cond(e) = en(i(c)) \wedge (\exists x \in \alpha : \neg en(x))$.
For $i \in K$: $inc(i) = \{q_i \in Q_i \mid q_i \text{ is incomplete}\}$.
If $p = e_1, ..., e_r$ is a path in G_{Sys}, then we put $cond(p) = \bigwedge_{i=1}^{r} cond(e_i)$.

In the next definition the notion of critical path is introduced. A critical cycle describes a situation where cyclic waiting of components could arise.

Definition 11
A path p in G_{Sys} is called critical, *if $(cond(p) \wedge \bigwedge_{i \in p} inc(i)) \not\equiv false$, where $i \in p$ means that node i is the start of some edge of p. A path p in $G_{Sys}(q)$ is called* critical *if $(cond(p) \wedge \bigwedge_{i \in p} inc(i))\ (q) \neq false$. A path that is not critical is called non-critical.*

Certain paths can immediately be singled out as non-critical.

Lemma 1
If $c \in L_1$ occurs $\mid c \mid$-times as a label on path p in G_{Sys}, where for any two edges $e = (i, c, k)$, $e' = (j, c, l)$ we have $i \neq j$, then $cond(p) \equiv false$. (Analogously for the label (c, α)).

Definition 12
Let p be a critical cycle in a finite successor-closed subgraph G_f of G_{Sys}, $q = (q_1, q_2, ...)$ a global state. p is said to be refutable, *if, whenever p lies in $G_f(q)$, where q_i is incomplete for every i, then there is a non-critical path $\hat{p}$ in $G_f(q)$ such that for every edge $e = (i, c, j)$, resp. $e = (i, (c, \alpha), j)$, on that path $en(i(c))(q_i)$ holds.*

There are some simpler but stronger conditions that guarantee refutability.

Lemma 2
Let p be a critical cycle in a finite successor-closed subgraph G_f of G_{Sys}. p is refutable, if one of the following conditions holds:

a) whenever p lies in $G_f^a(q)$ then there is a non-critical path $\hat{p}$ in $G_f^a(q)$ with $i \in \hat{p} \Rightarrow i \in p$.
b) whenever p lies in $G_f(q)$ then there is a non-critical path $\hat{p}$ in $G_f(q)$ such that for every edge $e = (i, c, j)$, resp. $e = (i, (c, \alpha), j)$, on that path $en(i(c))(q_i)$ holds if q_i is incomplete.

As the next theorem shows, a system is deadlock-free if there is a successor-closed subgraph of G_{Sys} that does not contain any critical cycle. One can show that a system satisfies this condition iff it satisfies the condition of [9]. In addition, the theorem states that, if there is no such subgraph, we have the option to check if there is a subgraph in which the critical cycles can be *refuted*. The second part of the theorem is more of theoretical interest and gives a characterization of of deadlock-free systems in terms of snapshots of the graph.

Theorem 1
Let Sys be an interaction system as above.

1) *If there is a finite nonempty successor-closed subgraph G_f of G_{Sys} such that every critical cycle in G_f is refutable, then Sys is deadlock-free.*
2) *Sys is deadlock-free iff $\forall q \in Q$ the following holds*
   a) *either $G_{Sys}(q)$ has a node with out-degree 0*
   b) *or there is $\alpha \in L_1$ resp. $(c, \alpha) \in L_2$ such that $\alpha = \{a_{i_1}, ..., a_{i_r}\}$, $a_{i_j} \in A_{i_j}$ and $G_{Sys}(q)$ has a simple cycle with the nodes $i_1, ..., i_r$ where all labels are α, resp. (c, α) and for every such cycle p $cond(p)(q) = false$.*

Proof: see Appendix

Remark 5
If $|c| \leq 2$ for all $c \in C$ then the condition in 1) can be tested in polynomial time. The theorem can be formulated analogously for the notion of P-deadlock-freedom.

Example 3 continued: G_{Sys} has no proper successor-closed subgraphs, so we have to check G_{Sys} for critical cycles. We discuss the case $n = 3$. In this case G_{Sys} is given in Figure 4.

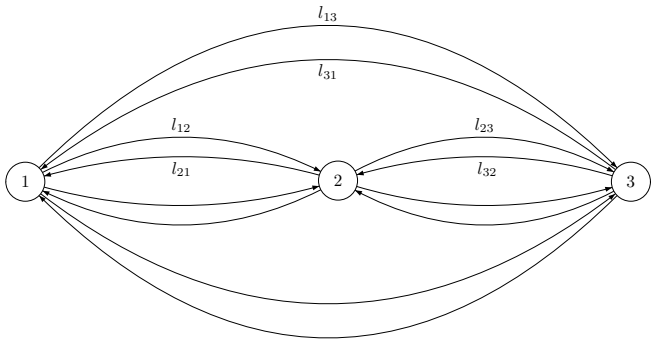

Fig. 4. The graph G_{Sys} for the scheduling example, $n = 3$

Here $l_{ij} = (conn_3^{ij}, \{finish_j\})$ where $conn_3^{ij} = \{resume_i, finish_j\}$. We have omitted the label $conn_g$ for better reading, so all edges without label carry the label $conn_g$. Let us consider the cycle

$$p = (1, (\{resume_1, finish_2\}, \{finish_2\}), 2), (2, (\{resume_2, finish_1\}, \{finish_1\}), 1).$$

This cycle is critical as in $q = (susp_1, susp_2, q_3)$ with $q_3 \in \{inac_3, wait_3, exec_3, susp_3\}$

$$cond(p) = (en(resume_1) \wedge \neg en(finish_2)) \wedge (en(resume_2) \wedge \neg en(finish_1))$$

is satisfied and q_i is incomplete for $i = 1, 2$. This cycle, however, can be refuted. If the last state is complete, i.e. $\neq susp_3$, then we consider the path $\hat{p} = (3, l_{32}, 2)$. The path $\hat{p}$ is non-critical and satisfies condition b) of Lemma 2. If the last state is incomplete, i.e. $= susp_3$ then the path $\hat{p} = (1, conn_g, 2)$ is non-critical as

$$cond(\hat{p}) = en(reset_1) \wedge (\neg en(reset_2) \vee \neg en(reset_3))$$

is false in the state $(susp_1, susp_2, susp_3)$. The other cycles in the graph are treated analogously.

The graph that we use to determine deadlock-freedom of a system can also be used to determine if a subsystem $K' \subseteq K$ may progress. For this we construct the restriction $IM[K']$ of the interaction model IM to K' as given in the **Appendix** and define when K' is controllable with respect to an interaction $\alpha \in I(C[K'])$, that is a potential partner for an interaction in $I(C) \setminus Comp$, which means that we may ensure that the subsystem defined by K' will be able to provide α, when it is needed.

Definition 13
Let $Sys = ((K, \{A_i\}_{i \in K}), IM, T)$ be an interaction system, $K' \subseteq K$. Let $Sys[K'] = ((K', \{A_i\}_{i \in K'}), IM[K'], T[K'])$ be the induced system where

$$T[K'] = (Q[K'], I(C[K']), \rightarrow'), \quad Q[K'] = \prod_{i \in K'} Q_i$$

is given by Definition 4.

A state $q \in Q[K']$ is called complete in $Sys[K']$ if $\exists q' \; \exists \alpha \in Comp[K'] \cup C[K'] : \; q \overset{\alpha}{\rightarrow}' q'$.
For $X \subseteq Q[K']$ define $pre(X) \subseteq Q[K']$ such that $q \in pre(X)$ if

a) $q \in Q[K']$ is complete in $Sys[K']$ then $\exists \alpha \in Comp[K'] \; \exists q' : \; q \overset{\alpha}{\rightarrow}' q' \wedge \; q' \in X$
b) $q \in Q[K']$ is incomplete in $Sys[K']$ then

$$\forall q' \in Q[K'] \; \forall \alpha \in (I(C[K']) \setminus Comp[K']) : \; (q \overset{\alpha}{\rightarrow}' q' \implies q' \in X)$$

For $Q_0 \subseteq Q[K']$, we denote by $PRE(Q_0)$ the least solution of $X = Q_0 \cup pre(X)$. $PRE(Q_0)$ is the set of states from which we can reach a state in Q_0 along a path in $T[K']$ by always performing complete interactions whenever a state is complete in $Sys[K']$.

Definition 14
Let Sys be a deadlock-free interaction system.
K is controllable with respect to $\alpha \in I(C)$, *if $PRE(en(\alpha)) = Q$.*
$K' \subseteq K$ is controllable with respect to IM, *if $\forall \alpha' \in I(C[K'])$*

$$(\exists \alpha \in I(C) \setminus Comp : \; (\alpha \cup \alpha' \in I(C) \implies$$
K' is controllable with respect to α' in the induced subsystem $Sys[K']$)).

We can now present a condition ensuring that a subsystem induced by K' may strongly progress, i.e. for every run σ we may at any point continue with a run σ' such that every component of K' will participate at some time in the run σ'.

Theorem 2
Let Sys be a deadlock-free interaction system. Let $K' \subseteq K$ finite or infinite.
K' may strongly progress in Sys, if the following two conditions hold

a) $\forall i \in K'$ *a finite successor-closed subgraph $G_{f,i}$ of G_{Sys} that contains i and does not contain any critical cycle.*
b) $\forall i \in K'$ $\forall \alpha \in C[K'']$ *owners(α) is controllable with respect to IM, where K'' is the set of components of $G_{f,i}$, and owners(α) = $\{k \in K \mid k(\alpha) \neq \emptyset\}$.*

Proof: see Appendix.

In an analogous way, we can treat the concept of may progress.

5 Composition of systems

5.1 Composition of Interaction Models and Systems

In this section, we explain how interaction systems can be constructed bottom up from smaller interaction systems. In contrast to other bottom up techniques composition is performed in such a way that the individual components remain visible after composition. The composition operator can be also used as a basis for a different approach to tackle the problem of NP-hardness of deciding the properties introduced above. In order to establish a desirable property for a composite system we might try to establish the property for the subsystems and infer the property for the composite system. In Section 5.2 we display some first conditions under which such a procedure can be applied.

Our view of composition is given in the following. Composition is determined on the level of the *interaction models*. We start with two (or more) disjoint interaction models, say $IM_i = (C_i, Comp_i)$ for finite component systems $(K_i, \{A_j\}_{j \in K_i})$, $i = 1, 2$. We then decide how these two models should be able to interact by providing a new set S of connectors that relate an interaction of one model with an interaction of the other. We always assume that S contains only **maximal** elements with respect to set inclusion, in analogy to the definition of a connector set. In addition, we may declare a set $BComp$ of new complete elements. For those we request that they **must** be *new* interactions in $I(S)$. This condition guarantees that by putting systems together, we do not interfere with the structure of the sub-models.

The following proposition explains how to compose in general two interaction models with respect to a given set S of new connectors and a set $BComp$ of new complete elements.

Notation 2
Let X, Y be sets of port sets. Then $X \bowtie Y := \{x \cup y \mid \ x \in X \wedge y \in Y\}$. maxel($X$) denotes the set of maximal elements of X with respect to set inclusion.

Proposition 1
Let $CS_1 = (K_1, \{A_i\}_{i \in K_1})$, $CS_2 = (K_2, \{A_i\}_{i \in K_2})$ be two component systems such that K_1, K_2 are disjoint and the elements in $\{A_i\}_{i \in K_1 \cup K_2}$ are pairwise disjoint. Let $IM_i = (C_i, Comp_i)$ be interaction models for CS_i, $i = 1, 2$. Let

– $S \subseteq I(C_1) \bowtie I(C_2)$ *be the set of new connectors,*
– $BComp \subseteq I(S) \cap (I(C_1) \bowtie I(C_2))$ *be the set of new complete elements.*

We put

– $C = maxel(C_1 \cup C_2 \cup S)$,
– $Comp = cl(Comp_1 \cup Comp_2 \cup BComp, I(C))$.

Then

a) C *is a connector set for* $(K_1 \cup K_2, \{A_i\}_{i \in K_1 \cup K_2})$ *with* $S \subseteq C$ *and* $I(C) = I(C_1) \cup I(C_2) \cup I(S)$
b) $IM = (C, Comp)$ *is an interaction model for* $(K_1 \cup K_2, \{A_i\}_{i \in K_1 \cup K_2})$.

Definition 15
Under the conditions of Proposition 1, the interaction model $IM = (C, Comp)$ *is said to be obtained by* composition *from* IM_1 *and* IM_2 *and we write*

$$IM = IM_1 \bigcup_{S, BComp} IM_2.$$

When composing interaction models IM_i and IM_j for some i, j using a set S of new connectors we use the notation S_{ij} to denote S, where it is understood that $S_{ij} = S_{ji}$. Analogously, $BComp_{ij}$ is used to denote the set of new complete elements and it is understood that $BComp_{ij} = BComp_{ji}$.

The composition of interaction models defined above is obviously commutative. One can show that it is **associative**. Associativity is an important property that allows composing systems incrementally without regarding the order in which subsystems are added. In process calculi, interaction is modelled by parallel operators that enforce restriction as in some versions of CSP [5] or a general parallel operator in combination with a restriction operator as e.g. in CCS. Associativity is generally not given for process calculus type parallel constructs.

Remark 6
The composition operator defined in Definition 15 is binary. If we want to compose systems from more than two subsystems then we can exploit the associativity of the operator. For example consider systems Sys_i, *where* $1 \leq i \leq 3$. *We want to compose these systems by introducing a connector* $c = \alpha_1 \cup \alpha_2 \cup \alpha_3$ *where* $\alpha_i \in I(C_i)$ *and another connector* $c' = \beta_1 \cup \beta_3$ *where* $\beta_i \in I(C_i)$. *To achieve this we first compose* Sys_1 *with* Sys_3 *using the connectors* $\alpha_1 \cup \alpha_3$ *and* c'. *Then we compose the resulting system with* Sys_2 *using the connector* $(\alpha_1 \cup \alpha_3) \cup \alpha_2 = c$. *Similarly we may handle the interactions that we want to declare complete.*

When the composition of the interaction models is defined, the interaction systems are now composed in a straightforward manner to yield a more complex system.

Definition 16
Let $Sys_i = ((K_i, \{A_j\}_{j \in K_i}), IM_i, T_i)$, $i = 1, 2$, *be interaction systems,* $S_{12} \subseteq I(C_1) \bowtie I(C_2)$ *a set of new connectors and* $BComp_{12} \subseteq I(S_{12}) \cap (I(C_1) \bowtie I(C_2))$, *then*

$$Sys_1 \Big|\Big|_{S_{12},\, BComp_{12}} Sys_2$$

is the system given by

$$Sys = ((K_1 \cup K_2, \{A_j\}_{j \in K_1 \cup K_2}), IM_1 \bigcup_{S_{12}, BComp_{12}} IM_2, T) \text{ where}$$

$T = (Q_1 \times Q_2, I(C_{12}), \rightarrow_{12})$ *and* $\rightarrow_{12}$ *is the least relation satisfying*

1. *if* $q_1 \stackrel{\alpha_1}{\rightarrow}_1 q_1'$ *then* $(q_1, q_2) \stackrel{\alpha_1}{\rightarrow}_{12} (q_1', q_2)$
2. *if* $q_2 \stackrel{\alpha_2}{\rightarrow}_2 q_2'$ *then* $(q_1, q_2) \stackrel{\alpha_2}{\rightarrow}_{12} (q_1, q_2')$
3. *if* $q_1 \stackrel{\alpha_1}{\rightarrow}_1 q_1'$, $q_2 \stackrel{\alpha_2}{\rightarrow}_2 q_2'$, $\alpha_1 \cup \alpha_2 \in I(S_{12})$ *then* $(q_1, q_2) \stackrel{\alpha_1 \cup \alpha_2}{\rightarrow}_{12} (q_1', q_2')$.

From the associativity for composition of interaction models, it is straightforward to see that the so defined parallel operator on interaction systems is associative and Remark 6 applies analogously.

5.2 Ensuring Deadlock-Freedom and Progress by Construction

In this section we rise the question under which conditions desirable properties of subsystems can be lifted to composed systems. As an example we treat deadlock-freedom and progress of a component. In the following, we consider two interaction systems $Sys_i = (CS_i, IM_i, T_i)$, $i = 1, 2$ that are composed by introducing a new set of connectors S_{12}. We assume that all conditions that are necessary to compose systems are fulfilled, as required in Proposition 1. We first consider deadlock-freedom.

Proposition 2
Let Sys_i $i = 1, 2$ *be interaction systems. If one of the following conditions is satisfied*

a) Let Sys_1 *be deadlock-free and* $C_1 \subseteq Comp_1$ *and let* $S_{12} \subseteq I(C_1) \bowtie I(C_2)$ *be arbitrary.*
b) Sys_1 *is deadlock-free and* $S_{12} \subseteq (I(C_1) \setminus C_1) \bowtie I(C_2)$
c) Sys_1 *is deadlock-free or* Sys_2 *is deadlock-free and*

$$S_{12} \subseteq (Comp_1 \bowtie (I(C_2) \setminus C_2)) \cup ((I(C_1) \setminus C_1) \bowtie Comp_2)$$

then

$$Sys_1 \Big|\Big| Sys_2$$
$$S_{12},\, BComp_{12}$$

is deadlock-free for any $BComp_{12} \subseteq I(S_{12}) \cap (I(C_1) \bowtie I(C_2))$.

Sketch of proof for b): in this case the connectors of the deadlock-free system Sys_1 *remain maximal after composition and as the complete elements of* Sys_1 *remain complete by definition every state in* $Sys_1 \Big|\Big|_{S_{12},\, BComp_{12}} Sys_2$ *offers a complete or maximal interaction in* $I(C)$.

These are only some examples for conditions that can be put on the level of the interaction models. Further conditions, as well as conditions imposing restrictions on the local transition systems, and conditions involving the graph criterion of Theorem 1 are being elaborated.

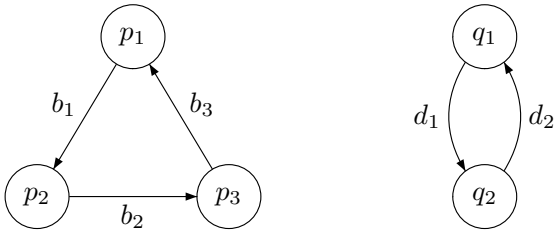

Fig. 5. Local Transition Systems

Example 4
We consider the following two component systems CS_1 and CS_2. The components of CS_1 are 1 and 2 those of CS_2 are 3 and 4. The port sets of 1 and 2 are given by $A_1 = \{a_1, a_2, a_3\}$ and $A_2 = \{c_1, c_2, c_3\}$. The port sets of the other two components are given by $A_3 = \{b_1, b_2, b_3\}$ and $A_4 = \{d_1, d_2\}$. We define $IM_i = (C_i, Comp_i)$ for CS_i where $C_1 := \{\{a_1, c_1\}, \{a_2, c_2\}, \{a_3\}, \{c_3\}\}$ and $Comp_1 := \{\{a_2\}, \{a_2, c_2\}, \{c_3\}\}$ respectively $C_2 := \{\{b_1\}, \{b_2, d_1\}, \{d_2\}, \{b_3\}\}$ and $Comp_2 := \{\{b_1\}\}$. The behavior of the second system Sys_2 is given by the two local transition systems of Figure 5. It is clear that Sys_2 is deadlock-free. Now let the behavior of the first system be given by any transition system that is labelled with interactions from $I(C_1)$. Composing the two systems according to $S_{12} := \{\{b_2, a_2, c_2\}, \{b_2, c_3\}\} \cup \{\{b_1, a_1\}, \{b_1, c_1\}\}$ and arbitrary $BComp_{12}$ yields a deadlock-free system according to Proposition 2 c).

We now consider the question under which conditions a component k that may progress in Sys_1 will still have this property when Sys_1 is composed with some other system. We present here two examples for conditions that are similar to those that ensure deadlock-freedom for the composite system.

Proposition 3
Let $Sys_i,\ i = 1, 2$, be interaction systems and let Sys_1 be deadlock-free and let $k \in K_1$ be a component that may progress in Sys_1. If one of the following conditions is satisfied

a) $S_{12} \subseteq (I(C_1) \setminus C_1) \bowtie I(C_2)$
b) $S_{12} \subseteq Comp_1 \bowtie I(C_2)$

then k may progress in $Sys_1 \Big|\Big|_{S_{12},\, BComp_{12}} Sys_2$ for any $BComp_{12} \subseteq I(S_{12}) \cap (I(C_1) \bowtie I(C_2))$.

Sketch of proof for a): the reason is that the maximal and complete interactions of Sys_1 are still so in $Sys_1 \Big|\Big|_{S_{12},\, BComp_{12}} Sys_2$.

6 Discussion and related work

The paper proposes results ensuring properties of component-based systems from properties of its interaction model and its components in the framework of interaction systems encompassing heterogeneous interaction presented in [9]. The following features of the framework are instrumental for developing our results:

- the separation of behavior and coordination, where coordination is not expressed by another behavior, but by an interaction model. We hope that the small examples illustrate the flexibility of this framework for coordinating a set of components at a high level of abstraction without interfering with the components' behaviors.
- the associativity of the framework which is the basis for well-defined incremental construction of systems.

An important motivation for introducing a clean theoretical framework for interaction and control is the hope that this will provide means for proving properties – which in general have to be shown on global models – in an incremental manner and as independently as possible from the behavior models of components. It has been shown in [13] that deciding deadlock-freedom in interaction systems is NP-hard. A similar result exists for liveness of a set of components [12]. This motivates the focus on conditions for establishing such properties. Our examples show the usefulness of these results. More such results are still needed, and can probably be obtained when some specificities of an application domain are exploited.

Related methods for showing absence of deadlocks or mutual waiting have been studied in various settings, for example in the context of data base transactions in a more restricted form [18,14] or operating systems for dynamic deadlock avoidance [17].

More recently, in [2], absence of interlocking has been studied for a framework of communicating processes where processes interact in all their interactions with the same fixed set of other processes. In [1] a condition under which a composed system is deadlock-free is given in a CSP-like setting. In [3] a condition for the deadlock-freedom of a component-system with two components is given.

Extended versions of our framework, including local variables of components and priority rules as an additional control layer, are presently being implemented. The implementation in the context of the Prometheus tool focusses on verification, in particular verification of SystemC specifications. A second implementation, called BIP, focusses on the efficient execution of systems and includes also timed specifications.

The work presented here shows some typical results that can be established in this framework. Further properties such as liveness, fairness and robustness as well as a result presenting a condition that can be tested in polynomial time and ensures liveness can be found in [6]. There is work in progress that is concerned with robustness and further exploits compositionality. Moreover first steps to incorporate probabilities have been taken in order to eventually be able to make quantitative propositions about the properties presented here. For example we want be able to prove statements like "with probability p no deadlock occurs". The notion of a component can be extended with various additional information including invariants [9], but also observability criteria and associated equivalence relations are of interest. Other possible interesting extensions concern introduction of time, as well as dynamic reconfiguration.

References

1. Robert Allen and David Garlan. A Formal Basis for Architectural Connection. *ACM Trans. Softw. Eng. Methodol.*, 6(3):213–249, 1997.

2. P.C. Attie and H. Chockler. Efficiently verifiable conditions for deadlock-freedom of large concurrent programs. In R. Cousot, editor, *proc. VMCAI'05*, volume 3385 of *LNCS*, pages 465–481, 2005.
3. Hubert Baumeister, Florian Hacklinger, Rolf Hennicker, Alexander Knapp, and Martin Wirsing. A Component Modell for Architectural Programming. In *Proceedings of FACS 2005*. ENTCS, 2005.
4. Luca de Alfaro and Thomas A. Henzinger. Interface automata. In *Proceedings of the 8th European Software Engineering Conference, Vienna*, pages 109–120, 2001.
5. N. Francez, C.A.R. Hoare, D.J. Lehmann, and W.P. de Roever. Semantics of nondeterminism, concurrency and communication. *Journal of Computer Science*, 19:290–308, 1979.
6. G. Gössler, S. Graf, M. Majster-Cederbaum, M. Martens, and J. Sifakis. An Approach to Modelling and Verification of Component Based Systems. In *Proceedings of the 33rd International Conference on Current Trends in Theory and Practice of Computer, Science SOFSEM07*, volume 4362 of *LNCS*, 2007.
7. Gregor Gößler and Joseph Sifakis. Component-based construction of deadlock-free systems. In *proceedings of FSTTCS 2003, Mumbai, India*, LNCS 2914, pages 420–433, 2003.
8. Gregor Gößler and Joseph Sifakis. Priority systems. In *proceedings of FMCO'03*, LNCS 3188, 2004.
9. Gregor Gößler and Joseph Sifakis. Composition for component-based modeling. *Sci. Comput. Program.*, 55(1-3):161–183, 2005.
10. Nancy A. Lynch and Mark R. Tuttle. An introduction to input/output automata. *CWI-Quarterly*, 2(3):219–246, September 1989.
11. Mila Majster-Cederbaum, Moritz Martens, and Christoph Minnameier. A Polynomial-Time Checkable Sufficient Condition for Deadlock-Freedom of Component-Based Systems. In *Proceedings of the 33rd International Conference on Current Trends in Theory and Practice of Computer, Science SOFSEM07*, volume 4362 of *LNCS*, 2007.
12. Moritz Martens, Christoph Minnameier, and Mila Majster-Cederbaum. Deciding Liveness in Component-Based Systems is NP-hard. Technical report TR-2006-017, Universität Mannheim, 2006.
13. Christoph Minnameier. Deadlock-Detection in Component-Based Systems is NP-hard. Technical report TR-2006-015, Universität Mannheim, 2006. also submitted for publication elsewhere.
14. Christos H. Papadimitriou. *The Theory of Database Concurrency Control*. Computer Science Press, 1986.
15. A. Pnueli. In transition from global to modular temporal reasoning about programs. In *Logics and Models for Concurrent Systems*. NATO, ASI Series F, Vol. 13, Springer Verlag, 1985.
16. Joseph Sifakis. A framework for component-based construction. In *3rd IEEE International Conference on Software Engineering and Formal Methods*, volume Key note talk, Koblenz, 2005.
17. A.S. Tanenbaum. *Modern Operating Sytems*. Prentice Hall, 2001.
18. Jeffrey D. Ullman. *Data Base and Knowledge-Base Systems, vol.1*. Computer Science Press, 1988.

7 Appendix

7.1 Subsystems

In order to be able to talk about subsystems, we define a notion of restriction of an interaction model to a subset of components.

Definition 17
Let $CS = (K, \{A_i\}_{i\in K})$ *be a component system,* $IM = (C, Comp)$ *an interaction model for* CS *and* $K' \subseteq K$. *Denote by*

$$C[K'] := \{c' \subseteq A[K'] \mid \exists\ c \in C :\ c' = c \cap A[K'] \wedge \nexists\ \bar{c} \in C :\ c' \subsetneq \bar{c} \cap A[K']\}$$

the restriction of the connectors in C *to* K' *where* $A[K'] = \bigcup\limits_{i\in K'} A_i$.
We put $Comp[K'] = I(C[K']) \cap Comp$.
If $C[K']$ *is a connector set for* $CS[K'] = (K', \{A_i\}_{i\in K'})$, *then the* induced interaction model *for* $CS[K']$ *is given by:* $IM[K'] := (C[K'], Comp[K'])$ *which is also called the* restriction *of* IM *to* K'.

Remark 7
Note that in the case of a system with infinitely many components it might occur that $C[K']$ *does not satisfy the conditions for a connector set. In particular,* $C[K']$ *might be empty for infinite systems. For finite systems* $C[K']$ *is always a connector set for* $CS[K'] = (K', \{A_i\}_{i\in K'})$ *and* $I(C[K']) = \{\alpha \cap A[K'] \mid \alpha \in I(C)\}$.

7.2 Proofs

Proof of Theorem 1
The proof of part 1) is an extension of the proof of the corresponding theorem in [9] if there is a successor-closed subgraph that does not contain any critical. In addition one has to show that refutability will do if there are critical cycles. We give here the proof of part 2):

Let Sys be deadlock-free and $q = (q_1, q_2, \ldots)$ a state in Q.
Case 1: There exists i such that q_i is complete.
Then the node i has out-degree 0 in $G_{Sys}(q)$.
Case 2: $\forall i\ q_i$ is incomplete.
As Sys is deadlock-free, there is some $\beta \in C \cup Comp\ \ q \xrightarrow{\beta} q'$ for some q'.
Case 2.1: β is a minimal element in $C \cup Comp$.
Case 2.1.1: $\beta = \{a_{i_1}, \ldots, a_{i_r}\} \in C,\ r > 1$
Hence $\beta \in L_1$, then choose $\alpha = \beta$. Hence (i_j, α, i_k) is an edge in $G_{Sys}(q)$, $j \neq k,\ j, k = 1, \ldots, r$. I.e. we obtain edges labelled with α between any two nodes in $\{i_1, \ldots, i_r\}$. In particular there is the cycle $i_1 \xrightarrow{\alpha} i_2 \ldots\ldots \xrightarrow{\alpha} i_r \xrightarrow{\alpha} i_1$. For any such cycle p between these nodes $cond(p)(q) = false$.
Case 2.1.2: $\beta = \{a_{i_1}, \ldots, a_{i_r}\} \in Comp \setminus C,\ r > 1$
Let c be a connector with $\beta \subset c$. Put $\alpha = \beta$. Then $(c, \alpha) \in L_2$. Hence we get edges $(i_j, (c, \alpha), i_k)$ in $G_{Sys}(q)$, $j \neq k,\ j, k = 1, \ldots, r$ and for all cycles p over all these nodes labelled with $(c, \alpha)\quad cond(p)(q) = false$.
Case 2.2: β is not minimal in $C \cup Comp$
Then there is some minimal $\alpha \in Comp$ with $\alpha \subset \beta$. As there is some transition $q \xrightarrow{\beta} q'$ for some q' in T, we also have a transition $q \xrightarrow{\alpha} q''$ for some q''. Let $\alpha = \{a_{i_1}, \ldots, a_{i_r}\},\ r > 1$. We now proceed as in Case 2.1.2.

Let *conversely* each state q satisfy a) or b). Let q be an arbitrary state. We have to show that there is some $\alpha \in C \cup Comp$ with $q \xrightarrow{\alpha} q'$ for some q'.
If a) is satisfied then let i be the node in $G_{Sys}(q)$ with out-degree 0. In this case q_i is complete and the statement is obvious.

If b) is satisfied then there is $\alpha \in L_1$ or $(c, \alpha) \in L_2$ such that $\alpha = \{a_{i_1}, ..., a_{i_r}\}$ whith $a_{i_j} \in A_{i_j}$ and $G_{Sys}(q)$ has a simple cycle with the nodes $i_1, ..., i_r$ where all labels are α, resp. (c, α) and for every such cycle we have $cond(p)(q) = \ false$.
Case 1: Label α. Let w.l.o.g. $i_1 \stackrel{\alpha}{\rightarrow} i_2 \ldots\ldots \stackrel{\alpha}{\rightarrow} i_r \stackrel{\alpha}{\rightarrow} i_1$ be a simple cycle with $cond(p)(q) = \ false$. Then there must be one edge e with $cond(e) = \ false$, hence for some j $(en(a_{i_j}) \wedge \exists x \in \alpha, x \neq a_{i_j} : \neg en(x))(q) = \ false$. But this edge is only in $G_{Sys}(q)$ if $en(a_{i_j})(q_{i_j}) = \ true$, hence $\alpha \in C$ can be performed in state q.
Case 2: Label (c, α): analogously.
Proof of Theorem 2:
One shows in a first step that, under the conditions of the theorem, every $k \in K'$ may progress in Sys. This can be done in a similar way as in [9] and is hence omitted here. In a second step we conclude that for finite K', K' may strongly progress and finally we show how to obtain the result for infinite K'. Let K' be finite, $K' = \{k_1, k_2 \ldots, k_n\}$. Let $\sigma = q_0 \stackrel{\alpha_0}{\rightarrow} q_1 \stackrel{\alpha_1}{\rightarrow} q_2 \ldots$ be a run of Sys and $n \in \mathbb{N}$. As k_1 may progress in Sys there exists σ' such that $\sigma(1) = \sigma_n \sigma'$ is a run of Sys and k_1 participates in some interaction of σ'. Let σ'_{l_1} be a prefix of σ' of length l_1 such that k_1 participates in some interaction of this prefix. Set $n_1 = n + l_1$ then there must be some σ'' such that $\sigma(2) = \sigma(1)_{n_1} \sigma''$ is a run of Sys and component k_2 participates in some interaction of σ''. As K' is finite this process terminates and we have a run in which finally every component participates in some interaction. We now assume that K' is countable. We consider the set R of all runs of Sys and introduce a metric $d : R \times R \rightarrow \{0, 1\}$ on R by $d(\sigma, \hat{\sigma}) = inf\{1/2^n : \ \sigma_n = \hat{\sigma}_n\}$. (R, d) is a complete metric space by standard arguments, hence every Cauchy-sequence converges in R. We now proceed as above and construct for every $m \in \mathbb{N}$ a run $\sigma(m)$ such that every element $\{1, 2, \ldots, m\}$ participates in some interaction of $\sigma(m)$ by maintaining the prefix that has participation of $\{1, \ldots, m\}$ when constructing the run that has participation of $\{1, \ldots, m+1\}$. In such a way we construct a Cauchy-sequence of runs $\sigma(1), \sigma(2), \ldots$. The limit of this sequence is the desired run that has the prefix σ_n and then contains for every $k \in K'$ an interaction in which k participates.