A Polynomial-Time Checkable Sufficient Condition for Deadlock-Freedom of Component-Based Systems

Mila Majster-Cederbaum[1], Moritz Martens[2], and Christoph Minnameier[3]

Institut für Informatik, Universität Mannheim, Germany
mcb@informatik.uni-mannheim.de, mmartens@informatik.uni-mannheim.de, cmm@informatik.uni-mannheim.de

Abstract. Interaction systems are a formal model for component-based systems. Combining components via connectors to form more complex systems may give rise to deadlock situations. Deciding the existence of deadlocks is NP-hard as it involves global state analysis. We present here a parametrized polynomial-time algorithm that is able to confirm deadlock-freedom for a certain class of interaction systems. The discussion includes characteristic examples and displays the role of the parameter of the algorithm.

1 Introduction

We consider a setting where components are combined via connectors to form more complex systems, see, e.g. [4], [6] and [7]. Each individual component i offers ports $a_i, b_i, .. \in A_i$ for cooperation with other components. Each port in A_i represents an action of component i. The behavior of a component is represented via a finite labeled transition system with starting state, where in each state there is at least one action available. Components are glued together via connectors, where each connector connects certain ports. In the global system obtained by gluing components (local) deadlocks may arise where a group of components is engaged in a cyclic waiting and will thus no longer participate in the progress of the global system (cf. [12]). It can be shown that testing the existence of local deadlocks is NP-hard [3]. We present here a parametrized polynomial-time algorithm that can confirm local deadlock-freedom for a certain class of systems. The algorithm is based on two ideas: first, a necessary condition for the existence of local deadlocks. If a component j is involved in causing a local deadlock in the reachable global state q then there must be two other components satisfying certain properties referring to their respective enabled actions in the state q. This is similar to an idea of [2] presented there for systems communicating via shared variables. The second idea is to consider an over-approximation of the set of reachable states: we consider the states that can be reached by projecting the state space to any subsystem of size d, where d is a parameter of the algorithm (and the degree of the polynomial describing the cost of the algorithm). If local deadlock-freedom cannot be verified, the

Jan van Leeuwen et al. (Eds.): SOFSEM 2007, LNCS 4362, pp. 888–899, 2007.

algorithm reports so, in which case one has to apply other methods to clarify the situation further, as e.g model-checking [9] or exploiting compositionality. We present a nontrivial example where our algorithm detects deadlock-freedom and where global state space analysis would indeed take exponential time. We discuss the role of the parameter d. In particular we present an example where increasing the value of d yields the desired result. The paper is organized as follows. Section 2 contains the basic definitions. Section 3 gives the necessary condition for the existence of local deadlocks and presents the algorithm and its analysis. Section 4 presents two examples. Section 5 refers to related work. Section 6 contains a conclusion and an outlook to future work.

2 Components, Interactions and Interaction Systems

2.1 The Model

We consider here *interaction systems*, a model for component-based systems that was proposed and discussed in detail in [4], [6] and [5]. An **interaction system** is a tuple $Sys = (K, \{A_i\}_{i\in K}, C, \{T_i\}_{i\in K})$[1], where K is the set of **components**. W.l.o.g. we assume $K = \{1, ..., n\}$. Each component $i \in K$ offers a finite set of **ports** or **actions** A_i for cooperation with other components. The port sets A_i are pairwise disjoint. Cooperation is described by *connectors*. A **connector** is a set of actions $c \subseteq \bigcup_{i\in K} A_i$, where for each component i at most one action $a_i \in A_i$ is in c. A **connector set** C is a set of connectors, such that every action of every component occurs in at least one connector of C and connectors are maximal w.r.t. set inclusion.

The local behavior of each component i is described by $T_i = (Q_i, A_i, \rightarrow_i, q_i^0)$. Q_i is the finite set of local states, the sets Q_i are pairwise disjoint. $\rightarrow_i \subseteq Q_i \times A_i \times Q_i$ is the local transition relation and $q_i^0 \in Q_i$ is the local starting state. We denote the maximal size $m = \max_{i\in K}(\max(|\rightarrow_i|, |Q_i|))$ of a local transition system T_i by m. Given a connector $c \in C$ and a component $i \in K$ we denote by $i(c) := A_i \cap c$ the **participation** of i in c. We identify a singleton set with its element.

For $q_i \in Q_i$ we define the set of **enabled actions** $ea(q_i) := \{a_i \in A_i \mid \exists q_i' \in Q_i, \text{s.t. } q_i \xrightarrow{a_i} q_i'\}$. We assume that the T_i's are non-terminating, i.e. $\forall i \in K\ \forall q_i \in Q_i\ ea(q_i) \neq \emptyset$.

The global behavior $T_{Sys} = (Q, C, \rightarrow, q^0)$ of Sys (henceforth called **global transition system**) is obtained from the behaviors of the individual components, given by the transition systems T_i and the connectors C 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 $(q_1, ..., q_n)$ and call them global states.

[1] The model in [4] is more general and distinguishes between connectors and complete interactions, but for the purpose of deadlock detection this distinction is irrelevant and we omit it for ease of notation.

- the relation $\rightarrow \subseteq Q \times C \times Q$, defined by
  $\forall c \in C\ \forall q, q' \in Q\ \ q = (q_1, ..., q_n) \xrightarrow{c} q' = (q'_1, ..., q'_n)$ iff
  $\forall i \in K\ \ (q_i \xrightarrow{i(c)}_i q'_i$ if $i(c) \neq \emptyset$ and $q'_i = q_i$ otherwise).
- $q^0 = (q^0_1, ..., q^0_n)$ is the starting state for *Sys*.

Let $q = (q_1, ..., q_n) \in Q$ be a global state. We say that some non-empty set $D = \{j_1, j_2, ..., j_{|D|}\} \subseteq K$ of components is in **deadlock** in q if $\forall i \in D\ \forall c \in C$: $c \cap ea(q_i) \neq \emptyset \Rightarrow \exists j \in D\ j(c) \not\subseteq ea(q_j)$.

A system has a **local deadlock** in some global state q if there is $D \subseteq K$, that is in deadlock in q. If $D = K$, the system is globally deadlocked. Hence a **global deadlock** is a special case of a local deadlock and we will henceforth simply speak of formation of deadlocks instead of local deadlocks. A system is *deadlock-free*, if there is no reachable state q and $D \subseteq K$, such that D is in deadlock in q.

Example
$Sys_1 = \{\{1, 2, 3, 4\}, \{A_i\}_{1 \leq i \leq 4}, C, \{T_i\}_{1 \leq i \leq 4}\}$, where the T_i's and A_i's can be seen from Figure 1, and $C = \{\{a_1, a_3\}, \{b_1, b_2\}, \{c_1, c_3\}, \{c_2, c_3\}, \{d_1, d_4\}, \{e_3, e_4\}, \{f_4\}\}$. The System starts in (q_1, q_2, q_3, q_4). If it chooses to perform the connector $\{d_1, d_4\}$ it reaches the global state (q'_1, q_2, q_3, q'_4). In this state, $D = \{1, 2, 3, 4\} = K$ is in (global) deadlock.

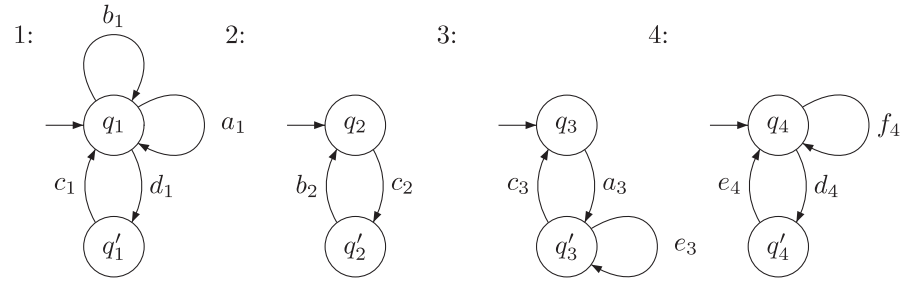

Fig. 1. The local transition systems for Sys_1

2.2 Some Technical Notions

Let $K_1 \subseteq K_2 \subseteq K$ and $U \subseteq \prod_{i \in K_2} Q_i$. Then $U \downarrow K_1$ consists of the **projection** of all states in U to the components in K_1.

Let $K' \subseteq K$. The transition system $T_{K'}$ induced by K' is given by $(Q_{K'}, C_{K'}, \rightarrow_{K'}, q^0_{K'})$, where $Q_{K'} := \prod_{i \in K'} Q_i$, $C_{K'} := \{c_{K'} = c \cap (\bigcup_{i \in K'} A_i) \mid c \in C\} \setminus \{\emptyset\}$, $\rightarrow_{K'}$ is defined analogously to $\rightarrow$ and $q^0_{K'} := q^0 \downarrow K'$.

That means, for the definition of $\rightarrow_{K'}$, we restrict connectors to actions of components in K'. This amounts to presuming (for reachability) that actions of components in $K \setminus K'$ are always available. This fact is directly associated to Remark 2.

Given a set $K' \subseteq K$ and the induced transition system $T_{K'}$ we denote by **reach-by-j**$(T_{K'}) \subseteq Q_{K'}$ the set of states that can be reached in $T_{K'}$ (starting in $q^0_{K'}$) in a way such that the last state transition affected component j. We first give a constructive (recursive) auxiliary definition of $reach(T_{K'}) \subseteq (Q_{K'} \times K')$ as follows:

- $(q^0_{K'} \xrightarrow{l}_{K'} q') \Rightarrow (\forall j \in K', \text{s.t. } q'_j \neq q^0_{K'j} : (q', j) \in reach(T_{K'}))$
- $((q, i) \in reach(T_{K'}) \wedge q \xrightarrow{l}_{K'} q') \Rightarrow (\forall j \in K', \text{s.t. } q'_j \neq q_j : (q', j) \in reach(T_{K'}))$

For $j \in K'$ we define reach-by-$j(T_{K'}) := \{q \in Q_{K'} \mid (q, j) \in reach(T_{K'})\}$.

Remark 1: For $T_{K'}$, $K' \subseteq K$ with $|K'| = d$, a reachability analysis can be performed in $O(m^d)$, which is an upper bound for the number of states in $T_{K'}$. We may store $reach(T_{K'})$ (i.e. the reachabilties plus the information about the components that may change their states in a preceding interaction) in a tabular of size $O(m^d \cdot n)$ which is also the required computation time. If $reach(T_{K'})$ is stored in a 2-dimensional array $(Q_{K'}, K')$, reach-by-$j(T_{K'})$ does not require any additional computation. It can directly be seen from the array's j-th column.

Remark 2: Let $Sys = (K, \{A_i\}_{i \in K}, C, \{T_i\}_{i \in K})$
$q \in$ reach-by-$j(T_{Sys}) \Rightarrow \forall K' \subseteq K,\ j \in K' : (q \downarrow K') \in$ reach-by-$j(T_{K'})$.
If a state q is reachable in the global transition system in a way such that the last state transition affected component j, then for every $K' \subseteq K$ that includes j, the state $q \downarrow K'$ is reachable in the corresponding subsystem in a way such that the last state transition affected component j.

We define for $Sys = (K, \{A_i\}_{i \in K}, C, \{T_i\}_{i \in K})$, $i, j, k \in K$ and $1 \leq d \leq n$ the set reach$_d$-by-$j(i, j, k) := \bigcap_{K' \subseteq K, \mathtt{s.t.}\, i,j,k \in K' \wedge |K'| = d}$ (reach-by-$j(T_{K'}) \downarrow \{i, j, k\}$).

Remark 3: $\forall 1 \leq d \leq n, (\text{reach-by-}j(T_{Sys}) \downarrow \{i, j, k\}) \subseteq \text{reach}_d\text{-by}-j(i, j, k)$
The projection of a reachable (by j) global state to $\{i, j, k\}$ is reachable (by j) in every subsystem of size d that includes i, j, k.

Example: We consider a system $Sys_3 = (K, \{A_i\}_{i \in K}, C, \{T_i\}_{i \in K})$, where $K = \{1, ..., 5\}$, $A_1 = \{a_1\}$, $A_2 = \{a_2, b_2, c_2\}$, $A_3 = \{b_3, d_3, e_3\}$, $A_4 = \{c_4, d_4\}$ and $A_5 = \{e_5\}$. $C = \{\{a_1, a_2\}, \{b_2, b_3\}, \{c_2, c_4\}, \{d_3, d_4\}, \{e_3, e_5\}\}$. The local transition systems are given in Figure 4 at the end of section 4. Consider the following exemplary reachabilities, where "-" stands for an arbitrary state of the corresponding component:

$(q_1, q'_2, q_3, q'_4, q_5) \in$ reach-by-2 (T_{Sys_3}); $\forall j \in K : (-, q'_2, q'_3, -, -) \notin$ reach-by-$j(T_{Sys_3})$; $(q_1, q'_2, q'_3) \in$ reach$_3$-by-2 $(1, 2, 3)$, whereas $(q_1, q'_2, q'_3) \notin$ reach$_4$-by-$2(1, 2, 3)$ even though $(q_1, q'_2, q'_3, q_5) \in$ reach-by-$2(T_{\{1,2,3,5\}})$, because $(q_1, q'_2, q'_3, -) \notin$ reach-by-$2(T_{\{1,2,3,4\}})$. proper subset of D is in deadlock in q then we speak of a minimal deadlock.

3 The Parametrized Polynomial Time Deadlock-Freeness Verification Algorithm

In this section we investigate the formation of deadlock situations in a system Sys. We assume that there is no deadlock[2] in the global starting state q^0. We derive a necessary condition for deadlocks that can be checked within subsystems and thus can be used to avoid exponential time complexity. Then, we present the parametrized verification algorithm in pseudocode and a short complexity analysis.

Lemma 1: Let q be a reachable global state for Sys and let $D \subseteq K$ be in deadlock in q, such that no proper subset of D is in deadlock in q. W.l.o.g. we assume that there is a transition sequence $q^0 \stackrel{c_0}{\rightarrow} q^1 \stackrel{c_1}{\rightarrow} ... \stackrel{c_{r-1}}{\rightarrow} q^r \stackrel{c_r}{\rightarrow} q$, s.t. no predecessor-state of q contains a deadlock. Then $\exists j \in D$, such that $q_j \neq q_j^r$ and the following conditions hold:

1) $\forall c \in C$, s.t. $c \cap ea(q_j) \neq \emptyset$ $\exists k \in D$ such that $k(c) \not\subseteq ea(q_k)$
(Every connector, j participates in, is blocked by some component k in D).
2) $\exists i \in D$ $\exists c \in C$, s.t. $c \cap ea(q_i) \neq \emptyset \wedge j(c) \not\subseteq ea(q_j)$
(In return, j blocks at least one enabled action of a component $i \in D$.)
Proof: See Appendix B.

We weaken *condition 1* by merely demanding the existence of a $c \in C$ and we apply *Remark 3* to formulate the following implication of the necessary condition in *Lemma 1* that can be observed in subsystems:

Corollary 1: Under the same assumptions as in Lemma 1, we may conclude:

$\exists i, j, k \in K$ $\exists q \in Q_{\{i,j,k\}}$ (namely the i and j from above, one of the k's in *condition 1* and the q from above projected to $\{i, j, k\}$), s.t. $\forall\ 1 \leq d \leq n$:
$q \in \text{reach}_d\text{-by-}j(i, j, k)$ and
1) $\exists c \in C_{\{i,j,k\}}$, s.t. $c \cap ea(q_j) \neq \emptyset\ \wedge k(c) \not\subseteq ea(q_k)$
(At least one interaction that j could participate in is blocked by k.)
2) $\exists c \in C_{\{i,j,k\}}$, s.t. $c \cap ea(q_i) \neq \emptyset \wedge j(c) \not\subseteq ea(q_j)$
(In return, j blocks at least one enabled action of a component i, as above.)

Given a subsystem $T_{\{i,j,k\}}$ and local states q_i, q_j we say "i (in q_i) *is blocked by* j (in q_j)" or "j (in q_j) *blocks* i (in q_i)"[3] iff $\exists c \in C_{\{i,j,k\}}$, s.t. $c \cap ea(q_i) \neq \emptyset \wedge j(c) \not\subseteq ea(q_j)$.

Given a subsystem $T_{\{i,j,k\}}$ and a state (q_i, q_j, q_k), we say that (q_i, q_j, q_k) is a *blocking chain* if i in q_i is blocked by j in q_j and j in q_j is blocked by k in q_k.

Hence (q_i, q_j, q_k) is a blocking chain if conditions 1 and 2 from *Corollary 1* hold.

[2] This is a natural assumption w.r.t. reasonable system design. Anyway, a check of this proposition would be possible within polynomial time.

[3] Actually, we should say "might be blocked" and "might block". However, as we check a necessary condition here and for ease of notation we use the shorter version.

The algorithm we present here, tries to confirm the negation of the necessary condition in the corollary and outputs "Sys is deadlock-free" if and only if it is successful in doing so. Otherwise, it outputs "Sys might contain deadlocks".

```
1: PROCEDURE deadlock-freedom-verifier(Sys, d)
2: for all i, j, k ∈ K do
3:    compute reach_d-by-j(i, j, k)
4: end for
5: for all i, j, k ∈ K do
6:    for all (q_i, q_j, q_k) ∈ reach_d-by-j(i, j, k) do
7:       if (q_i, q_j, q_k) is a blocking chain then
8:          write("Sys might contain deadlocks");
9:          break;
10:      end if
11:   end for
12: end for
13: write("Sys is deadlock-free")
14: END deadlock-freedom-verifier
```

The computation of reach$_d$-by-$j(i,j,k)$ for a subsystem $\{i,j,k\}$ (line 3) can be performed in $O(m^d \cdot n) \cdot (n^{d-3})$, as mentioned in *Remark 1.* The loop (2-4) is performed n^3 times, so we have an overall complexity of $O(m^d \cdot n^{d+1})$ for the reachability analyses.

The check, whether a state (q_i, q_j, q_k) is a blocking chain (line 7) can be performed in $O(|C| \cdot m)$, as we compare sets (of size ≤ 3) in $C_{\{i,j,k\}}$ elementwise with enabled actions in local transition systems.

The loop (6-11) is performed up to m^d times and the surrounding loop (5-12) is performed n^3 times. So 5-12 takes $O(m^{d+2} \cdot n^3 \cdot |C|)$ and this yields an overall complexity of $O(m^{d+2} \cdot n^{d+1} \cdot |C|)$.

4 Applicability

In the following, we present two example systems and apply our algorithm. We verify deadlock-freedom for a complex parametrized example system $Sys_2(y,x)$ and discuss how and why our algorithm is able to handle the example, even for arbitrarily large x, y with $d = 3$, i.e. observing subsystems of size 3 only. Then, we give an example system Sys_3 that can be proven deadlock-free with $d = 4$ but not with $d = 3$.

Trilateration is a method to determine the relative positions of an object on a surface, using triangulation of the surface. To accurately and uniquely determine the relative location of an object on a surface using trilateration, 3 reference points (in this case the vertices of the triangle surrounding the object) are needed.

Let us imagine a system of n transmitting stations that divide a surface into triangles, using an odd number y of rows and an odd number x of columns. (See Figure 2.) Three transmitting stations that form a triangle can cooperate in order to determine the position of an object within the triangle.

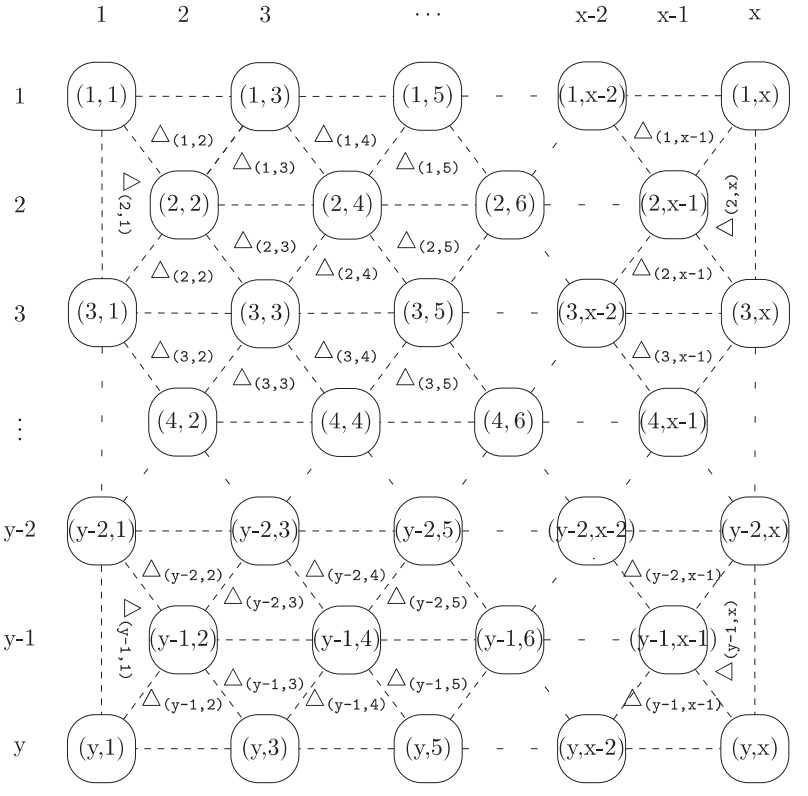

Fig. 2. An area divided into triangles $\triangle_{(a,b)}$ by transmitting stations (u, v)

That means, every transmitting station (u, v) can participate in a job (i.e. a trilateration) in one of its (up to) six adjacent triangle-areas at a time or participate in a maintenance together with the other ($\lfloor x/2 \rfloor$ or $\lfloor x/2 \rfloor - 1$) stations on the same horizontal line. Each transmitting station is a component (u, v) in our model and offers actions of type start, perform and end a cooperation in a triangle (a, b), which are abbreviated by s-c(u,v,a,b), p-c(u,v,a,b) and e-c(u,v,a,b), respectively. Also, each component (u, v) offers actions to start, perform and end a maintenance, which are abbreviated by s-maint(u,v), p-maint(u,v) and e-maint(u,v), respectively. The system is described by $Sys_2(y, x) = (K, \{A_{(u,v)}\}_{(u,v)\in K}, C, \{T_{(u,v)}\}_{(u,v)\in K})$, where:

$K = \{(2u+1, 2v+1) \mid 0 \leq u \leq \frac{y-1}{2}, 0 \leq v \leq \frac{x-1}{2}\}$
$\quad \cup \{(2u, 2v) \mid 1 \leq u \leq \frac{y-1}{2}, 1 \leq v \leq \frac{x-1}{2}\}$
$A_{(u,v)} = \{$s-c(u, v, a, b),p-c(u, v, a, b),e-c$(u, v, a, b) | \triangle_{(a,b)}$ is a triangle adjacent to $(u, v)\}$ $\cup$ $\{$s-maint(u, v),p-maint(u, v),e-maint$(u, v)\}$
C : For each $op \in \{$s-c,p-c,e-c$\}$ we include the connectors $\{op(u_1, v_1, a, b), op(u_2, v_2, a, b), op(u_3, v_3, a, b)\}$, where $(u_1, v_1), (u_2, v_2), (u_3, v_3)$ are vertices of $\triangle_{(a,b)}$.

Also, for $op \in \{$s-maint,p-maint,e-maint$\}$, we include the connectors
$\{op(u_1, 1), op(u_1, 3), ..., op(u_1, x)\}$, and
$\{op(u_2, 2), op(u_2, 4), ..., op(u_2, x-1)\}$ where
u_1 resp. u_2 ranges over the odd resp. even numbers in $\{1, ..., y\}$.

The $T_{(u,v)}$'s are depicted in Figure 3. Note that the transmitting stations at the margin of the area do not have 6 but less triangles to participate in, so Figure 3 is exemplary only.

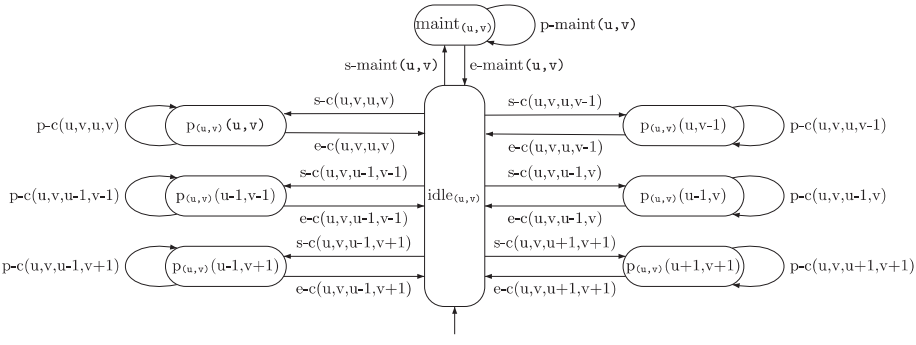

Fig. 3. The local transition system for some (non-marginal) transmitting station (u, v)

In the following, we prove that the algorithm is indeed able to verify (for arbitrarily large x, y) that $Sys_2(y, x)$ is deadlock-free, by showing that no subsystem $T_{\{i,j,k\}}$ of components $i, j, k \in K$ will ever reach a state (q_i, q_j, q_k), such that (q_i, q_j, q_k) is a blocking-chain:

Remark 4: Let a component $l_1 \in \{i, j, k\}$ be in its maint_{l_1}- (or in its p_{l_1}(a,b)-) state. In this case, l_1 offers its p-maint and e-maint (or p-c(a,b) and e-c(a,b)) actions. For $l_2 \in \{i, j, k\}$ to block l_1, l_2 must possess an action that occurs in a connector together with one of the actions offered by l_1, i.e. l_2 has to share a line with l_1 (or be one of the vertices of $\triangle_{(a,b)}$). However, as l_2 is observed in $T_{\{i,j,k\}}$ it must have gone to its maint_{l_2}- (or to its $p_{l_2}(a, b)$-) state conjointly with l_1 and thus offer the demanded action.

Now assume that there is a state (q_i, q_j, q_k), that is a blocking-chain:

Due to *Remark 4*, we may assume $q_i = \text{idle}_i$. But then, for q_j to block q_i, we have $q_j \neq idle_j$. But by *Remark 4*, we know that j in $q_j \neq \text{idle}_j$ cannot be blocked by k in any q_k.

We showed that our algorithm can verify deadlock-freedom for the trilateration example in polynomial time. Note, that the example is a non-trivial system that could easily be modeled to contain deadlocks, e.g. $(3, 3)$,$(2, 4)$,$(3, 5)$ could wait for each other when $(3, 3)$ is in a state where it wants do a job in $\triangle_{(2,3)}$ while $(2, 4)$ wants to do a job in $\triangle_{(2,5)}$ and $(3, 5)$ wants to do a job in $\triangle_{(3,4)}$. So

first, it is not obvious by specification that the implementation will be deadlock-free. Second, the number of reachable global states of the system is exponential (in n)[4]. Hence any algorithm that checks some condition for every global state would need time exponential in n. Third, the system scale is variable and it may contain arbitrarily large connectors (the maintenance connectors' size is linear in x). Nevertheless to verify deadlock-freedom it suffices to choose the parameter $d = 3$, i.e. to observe subsystems of size 3 only.

We are now going to investigate an example of a deadlock-free system Sys_3, for which our algorithm is not able to confirm deadlock-freedom when we observe subsystems of size 3. However, when observing subsystems of size 4, the algorithm yields the desired result.

Consider Sys_3, introduced at the end of section 2, for which the T_i's are given in Figure 4.

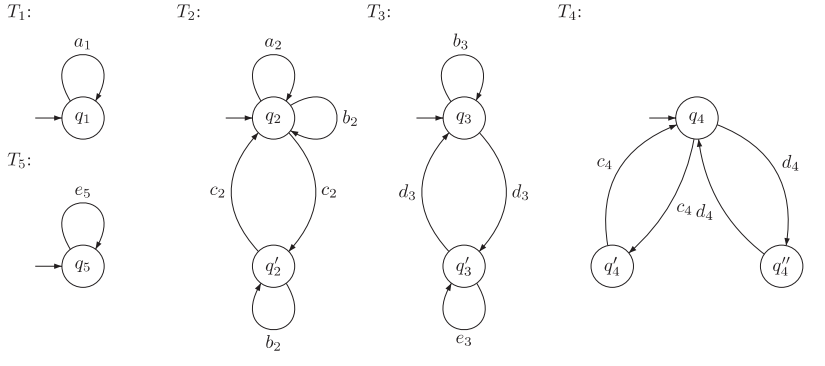

Fig. 4. The local transition systems T_i for Sys_3

When observing the subsystem $T_{\{1,2,3\}}$, we find $(q_1, q'_2, q'_3) \in \text{reach}_3\text{-by-2}\ (1, 2, 3)$ where 1 is blocked by 2, which is, in turn, blocked by 3. However, no corresponding global state is accessible in the global system because the communication with component 4 prevents 2 and 3 from reaching q'_2 and q'_3 simultaneously.

The problem, of course, is the lack of observation of component 4. If we apply the algorithm with $d = 4$, we are indeed able to verify deadlock-freedom: The relation $R = \{(q_1, q'_2), (q_2, q'_3), (q_2, q''_4), (q'_2, q''_4), (q'_2, q'_3), (q_3, q'_4), (q'_3, q'_4), (q_5, q_3)\}$ includes all pairs (q_i, q_j), where i in q_i is blocked by j in q_j. As a result, the set of possible blocking chains is $BC = \{(q_1, q'_2, q''_4), (q_1, q'_2, q'_3), (q_2, q'_3, q'_4), (q'_2, q'_3, q'_4), (q_5, q_3, q'_4)\}$. As stated at the end of section 2, $(q_1, q'_2, q'_3) \notin \text{reach}_4\text{-by-2}(1, 2, 3)$ and corresponding propositions hold for all other states in BC. The example also displays that it can be crucial to check whether a state is reachable by a state transition that affects a certain component:

[4] Proof: See Appendix A.

Note that (q_5, q_3, q_4'), where 5 is blocked by 3 and 3 is blocked by 4 is indeed reachable in both subsystems of size 4 that include the components 3, 4 and 5, but it is not reachable by an interaction that causes a local state change of the medium component 3, so the state's reachability alone will not affect the algorithm's success.

5 Related Work

There exist several approaches to solve the problem of deadlock-detection. In [8] and [4], sufficient conditions for verifying deadlock-freedom were given, but both without cost analyses. Then, based on [1] Attie and Chockler gave a sufficient condition along with a polynomial time algorithm to verify deadlock-freedoom for Parallel Processes in [2]. However, their framework differs significantly from the one discussed here. For one thing, it is totally interleaving, i.e. only one local transition system may change its state at a time, whereas in interaction systems [4], [6], [7] arbitrarily many (bound by the size of the largest connector) local transition systems may change their states simultaneously. Furthermore, actions in [2] are guarded commands, where the guards are conjunctions of predicates of (local) states. That means, it is sufficient as well as necessary for an action to be blocked that at least one communication partner is in a local state that does not meet the predicate. Things are different in component-based systems where an action can occur in different connectors and thus can be blocked by a whole set of components (due to the combination of their current local states). Also, [2] compute the set of reachable states of a component j as the union instead of the intersection of the reachable subsystems projected to j, i.e. they do not, like we do, take advantage of the fact that a (globally) reachable state's projection must be reachable in all subsystems containing the set on which we project. Finally the algorithm they present lacks parametrization which can be critical for success as example 2 shows.

6 Conclusion and Future Work

Our algorithm is (even with $d = 3$) able to handle the complex trilateration example Sys_2 regardless of the choice of the parameters x, y. That means it can handle arbitrarily large connectors and an exponential size reachable state space. Our algorithm profits from (see Sys_2), but is not dependent on (see Sys_3 with $d = 4$) symmetric constructs which are discussed in [10] and [11]. Sys_3 displays that the problem of "inherent information" can prevent our algorithm from verifying deadlock-freedom. This fact is, of course, not surprising as in [3] it is shown that the problem of deadlock-detection in component-based systems is NP-hard. Nevertheless the existence of non-trivial examples that cannot be verified in polynomial time by algorithms based on global state space exploration displays the benefit of the presented algorithm. As far as future work is

concerned, we will further investigate the applicability of the algorithm and the interesting question of the connection between connector sizes and the size d of subsystems one has to observe. Also, we are working on a generalization of the algorithm, in order to cover a greater class of systems (still maintaining the polynomial time bounds, of course).

References

1. Attie, P. and Emerson, A.: Synthesis of Concurrent Systems with Many Similar Processes. ACM TOPLAS **20** 1 (1998) 51–115
2. Attie, P. and Chockler, H.: Efficiently Verifiable Conditions for Deadlock-Freedom of Large Concurrent Programs. LNCS **3385** (2005) 465–481
3. Minnameier, C.: Deadlock-Detection in Component-Based Systems is NP-Hard. Technical Report TR-2006-015 (2006)
4. Gössler, G. and Sifakis, J.: Component-Based Construction of Deadlock-Free Systems. FSTTCS, LNCS **2914** (2003) 420-433
5. Gössler, G. and Sifakis, J.: Composition for Component-Based Modeling. Sci. Comput. Program. **55** 1-3 (2005) 161-183
6. Sifakis, J.: A Framework for Component-based Construction. Keynote Talk, SEFM (2005) 293–300
7. Gössler, G., Graf, S., Majster-Cederbaum, M., Martens, M., and Sifakis, J.: An Approach to Modelling and Verification of Component-based Systems. Accepted for SOFSEM 2007.
8. Aldini, A. and Bernardo, M.: A General Approach to Deadlock-Freedom Verification for Software Architectures. FM 2003, LNCS **2805** 658-677
9. Geilen, M.: Non-Exhaustive Model-Checking in Component Based Systems. Journal of Systems Architecture – The Euromicro Journal (2000)
10. Arons, T., Pnueli, A., Ruah, S., Xu, J., and Zuck, L.D.: Parameterized Verification with Automatically Computed Inductive Assertions. CAV (2001) 221-234
11. Clarke, E., Enders, R., Filkorn, T., and Jha, S.: Exploiting Symmetry in Temporal Logic Model Checking. FMSD **9** 2 (1996)
12. Tanenbaum, A.: Modern Operating Systems, 2nd ed. Prentice Hall (2001)

Appendix

A Exponential Size Global State Space of Sys_2

We show that the reachable global state space of the trilateration example is of exponential size. The vertices of any triangle can conjointly go into their $\triangle$-states, (given that none of the vertices is already in cooperation in another triangle it belongs to). That means, there are at least as many global states as there are ways to mark subsets of triangles, such that no pair of adjacent triangles is marked:

For systems with $x > 3$ and $y > 3$, the number of triangles exceeds the number of components. Hence, for sufficiently large examples with n components, we

have at least n triangles. Whenever we mark a triangle, the triangle itself plus the surrounding triangles may no longer be marked and so we always remove 13 (1 triangle + 12 neighbor triangles) from the set of triangles that are left for marking. It might happen that we mark a triangle which is next to a triangle which has already been removed from the set of left triangles, or that we mark a triangle which is at the margin of the trilateration area and thus doesn't have 12 neighbors, but in both cases we will nevertheless decrease the number of left triangles by 13 and thus achieve a lower bound for the number of triangles left. Hence, for $n \mod 13 = 0$ we have at least $n \cdot (n-13) \cdot (n-26) \cdot ... \cdot 13$ ways to mark as many triangles as possible, one after another. As the order in which we mark the triangles is not of concern, we have to divide by $(n/13)! = (n/13) \cdot (n/13-1) \cdot ... \cdot 2 \cdot 1$. Thus we have at least:

$$\frac{n \cdot (n-13) \cdot ... \cdot 26 \cdot 13}{(n/13) \cdot (n/13-1) \cdot ... \cdot 2 \cdot 1} = \frac{n}{n/13} \cdot \frac{n-13}{(n-13)/13} ... \cdot 13 \cdot 13 = 13^{(n/13)} \approx 1,22^n$$

B Proof of Lemma 1

For a reachable state $\tilde{q}$ there is a global transition sequence $s \xrightarrow{c_0} q_1 \xrightarrow{c_1} ... \xrightarrow{c_{r-1}} q_r \xrightarrow{c_r} \tilde{q}$. If $\tilde{q}$ is the first deadlocked state in this sequence let $q := \tilde{q}$, else let q be the first deadlocked state in the sequence and let $D \subseteq K$ be in deadlock for $q = (q_1, ..., q_n)$, such that no proper subset of D is in deadlock in q.

If neither of the components in D changed their local states in the interaction that lead to q, then D would have been a deadlock in the preceding state already, which would be a contradiction to our assumptions. So we have $\exists j \in D$, such that $q_j \neq q_j^r$.

It remains to show conditions 1 and 2:

Condition 1 follows directly from the definition of deadlocks as $j \in D$. Assume that *condition 2* does not hold, i.e. j in q_j does not block any other component $i \in D$. Then $D \backslash \{j\}$ would be in deadlock in q in contradiction to our assumptions.