Natural Computing (2024) 23:269–284
https://doi.org/10.1007/s11047-024-09974-5

Variants of distributed reaction systems

Erzsébet Csuhaj-Varjú[1] · György Vaszil[2]

Accepted: 16 January 2024 / Published online: 25 February 2024
© The Author(s) 2024

Abstract

A distributed reaction system consists of a finite set of reaction systems that either interact with a common environment or interact with each other by communicating products or reactions. A reaction system is a well-known qualitative formal model of interactions between biochemical reactions. A reaction is a triplet of nonempty sets representing chemicals, called the set of reactants, the set of inhibitors, and the set of products. A reaction corresponds to a chemical reaction performed on a set of chemicals, and a reaction system is a finite nonempty set of reactions. In this paper, we examine two variants of distributed reaction systems. We introduce the notion of a distributed reaction system with communication by request (a qDRS for short), where sets of products are communicated between the component reaction systems by queries. First, we show that every qDRS can be represented by a reaction system. After that we compare distributed reaction systems with communication by request to extended distributed reaction systems (EDRSs), models that were introduced in a previous paper. We prove that extended distributed reaction systems, where a context automaton provides input for the component reaction systems, simulate distributed reaction systems with communication by request and distributed reaction systems with communication by request simulate special variants of extended distributed reaction systems. Furthermore, we assign languages to these two variants of distributed reaction systems. We prove that the class of agreement languages of extended distributed reaction systems is equal to the class of languages of nondeterministic multihead finite automata and the agreement language of every distributed reaction system with communication by request is an element of a certain subregular language class.

Keywords Reaction system · Distributed reaction systems · Communication by request · Multihead finite automata

Mathematics Subject Classification 68Q07 · 68Q09 · 68Q45

1 Introduction

Reaction systems, introduced in 2004, are well-known qualitative models of biochemical interactions Ehrenfeucht et al. (2004). A reaction represents a chemical reaction and it is a triple of finite non-empty sets of objects: the set of reactants, the set of inhibitors, and the set of products. The set of reactants and the set of inhibitors are disjoint. A reaction system is a finite set of reactions.

A reaction can be performed on a set of reactants if each reactant of the reaction is present and each inhibitor is absent in the given reactant set. When a reaction is performed, then the set of its reactants is changed to the set of its products. All reactions of the reaction system that can be performed on a given set of reactants have to be performed in parallel. Those reactants that are not involved in any reaction, disappear from the set of reactants.

In the last two decades, several properties of reaction systems have been studied and several extensions have been introduced. Functions defined by reaction systems, properties of state sequences of reaction systems, the effect of bounded resources, and connections to propositional logic were examined in Ehrenfeucht et al. (2011); Salomaa (2013, 2012). Reaction systems can also be used as a modeling framework. For checking temporal properties of

✉ Erzsébet Csuhaj-Varjú
csuhaj@inf.elte.hu

György Vaszil
vaszil.gyorgy@inf.unideb.hu

1 Faculty of Informatics, ELTE Eötvös Loránd University, Pázmány Péter sétány 1/c, Budapest 1117, Hungary

2 Faculty of Informatics, University of Debrecen, Kassai út 26, Debrecen 4032, Hungary

reaction systems, a temporal logic was introduced in Meski et al. (2015). Biologically inspired properties of reaction systems have been examined in Azimi et al. (2014, 2016); Azimi (2017)

Reaction systems can also be organized in a distributed communication framework. In this case, reactions are located in nodes of virtual graphs or, simply, an *n*-tuple of reaction systems is given. The reactions in these systems work in a synchronized manner and interact with each other by distribution and communication protocols. Such interaction can take place either by obtaining input from a separate environment or by communicating reactants, products, or reactions to each other. Examples of such constructs are distributed reaction systems Meski et al. (2019), extended distributed reaction systems Ciencialová et al. (2022, 2023), networks of reaction systems Bottoni et al. (2020), and communicating reaction systems with direct communication Csuhaj-Varjú et al. (2020). Networks of reaction systems, distributed reaction systems, and communicating reaction systems were related in Aman (2022, 2023).

An extended distributed reaction system (the EDRS for short), introduced in Ciencialová L et al. (2022, 2023), is a slightly modified variant of the distributed reaction system model studied in Meski et al. (2019). An EDRS consists of a finite number of reaction systems defined over a common background set (set of objects). These reaction systems operate in a synchronized manner. In each step, a so-called context automaton provides a set of reactants for each reaction system which is added to the current reactant set (the current state) of this reaction system. This set of reactants is called a context. After that, all enabled reactions are performed on the obtained new set of reactants (the union of the original reactant set and the context) and the interactive process is going to be repeated.

In Csuhaj-Varjú et al. (2020), two variants of communicating reaction systems with direct communication (cdcR systems for short) were introduced, namely, the cdcR(p) system where products are communicated and the cdcR(r) systems where reactions are communicated. In the case of cdcR(p) systems, after performing the enabled reactions on its current reactant set, the component sends copies of certain products to certain target components. The target nodes and the products to be communicated are given together with the reaction. Thus, a computation step consists of a reaction and after that a communication. In the case of cdcR(r) systems, if a reaction is successfully performed by a component, then its copies are sent to predefined target components. Notice that these variants of distributed reaction systems realize communication by command, i.e., by performing the reactions, the components initiate the sending of specific products/reactants to other components.

In this paper, we introduce a new variant, called a distributed reaction system with communication by request (a qDRS for short). In this case, after performing the enabled reactions, the component may request the state of some other components, given together with the reaction. This query is represented by query symbols, each component is assigned to one query symbol and vice versa. Provided that the components addressed by a query have no query symbol in their states, a copy of the products in the state of the component is sent to the component that issued the query. This communication process continues until the state of every component is query-free.

If the communication process does not end with only query-free states, then the computation aborts. This model was inspired by parallel communicating grammar systems Păun and Kari (1989); Csuhaj-Varjú et al. (1994), a grammatical framework for distributed communicating systems of Chomsky grammars. In the case of parallel communicating grammar systems, the state of a component corresponds to the string generated by a grammar and the performing of a reaction corresponds to the application of a production.

In this paper, we examine distributed reaction systems with communication by request and extended distributed reaction systems and their relations to each other.

We prove that every distributed reaction system with communication by request can be represented by a reaction system. More precisely, to a given qDRS we can give a reaction system such that the sequence of its query-free states can be obtained by a simple mapping from the state sequence of the reaction system. Analogous results were presented for cdcR(p) systems and cdcR(r) systems in Csuhaj-Varjú et al. (2020). The result implies that the power of qDRS does not exceed the boundaries of the power of reaction systems.

Next, we show that to every qDRS a simulating EDRS can be given, i.e. their state sequences correspond to each other (they are almost the same). According to the reverse direction, to every EDRS of a certain restricted type, a simulating qDRS can be constructed. For the general case, such a simulation result cannot hold since, contrary to reaction systems and qDRS, extended distributed reaction systems can also be non-deterministic (when their context-automaton is non-deterministic).

In two previous papers Ciencialová L et al. (2022, 2023), languages were assigned to extended distributed reaction systems and representations of well-known language classes were presented with these models (the class of right-linear simple matrix languages and the class of recursively enumerable languages).

In this paper, we define the agreement language of the EDRS. We show that the class of agreement languages of extended distributed systems is equal to the class of languages of multihead nondeterministic finite automata. We discuss the languages of qDRS and we state that the agreement language of any distributed reaction system with communication by request is in a certain subregular language class.

Our paper is organized as follows. In Sect. 2, we present the basic notions concerning reaction systems and distributed reaction systems. In Section 3, we present the notion of a distributed reaction system with communication by request and statements concerning qDRS, including the comparisons to extended distributed reaction systems. In Section 4, we introduce the notion of the agreement language of an EDRS and prove the equality of the class of agreement languages of extended distributed systems and the class of languages of multihead nondeterministic finite automata. We also discuss the agreement language of qDRS. We close the paper with conclusions and suggest topics for research.

2 Preliminaries

Throughout the paper, we assume the reader to be familiar with the basics of formal language theory as presented, for example, in Hopcroft et al. (2006).

2.1 Reaction systems

In this subsection, we recall the basic notions concerning reaction systems, introduced in Ehrenfeucht et al. (2004); Ehrenfeucht and Rozenberg (2007).

Let S be a finite non-empty set. A triplet $a = (R, I, P)$ where R, I, P are nonempty subsets of S and $R \cap I = \emptyset$ is called a reaction in S. The set S is called the background set, R is called the set of reactants (or the reactant set), I is the set of inhibitors (or the inhibitor set), and P is called the set of products (or the product set) of reaction a; the elements of S are called objects or molecules.

A reaction system is an ordered pair $\mathcal{A} = (S, A)$, where A is a non-empty set of reactions in S.

Reaction systems function by performing their reactions on a nonempty subset of the background set that is called the current state of the reaction system. Let S be a background set, let $X \subseteq S$, and let $a = (R_a, I_a, P_a)$ be a reaction in S. Then, a is enabled by X, denoted by $en_a(X)$, if $R_a \subseteq X$ and $I_a \cap X = \emptyset$ holds. The result of a on X, denoted by $res_a(X)$, is defined by $res_a(X) = P_a$ if $en_a(X)$, and $res_a(X) = \emptyset$, otherwise.

The effect of a set of reactions on a state is cumulative. Let $\mathcal{A} = (S, A)$ be a reaction system with background set S and let $X \subseteq S$. The subset of reactions of A enabled by X, denoted by $en_A(X)$, is defined as $en_A(X) = \{a \in A \mid en_a(X)\}$, and the result of A on X, denoted by $res_A(X)$, is $res_A(X) = \{res_a(X) \mid a \in A\}$.

The behavior of the reaction system is described by an interactive process. Let $\mathcal{A} = (S, A)$ be a reaction system. An interactive process in $\mathcal{A}$ is a pair $\pi = (\gamma, \varphi)$ of finite sequences such that $\gamma = C_0, C_1, \ldots, C_n$, $\varphi = D_1, \ldots, D_n$ with $n \geq 1$, where $C_0, \ldots, C_n, D_1, \ldots, D_n \subseteq S$, $D_1 = res(A, C_0)$, and $D_i = res(A, D_{i-1} \cup C_{i-1})$ for each $2 \leq i \leq n$.

The sequences $C_0, \ldots, C_n$, and $D_1, \ldots, D_n$ are the context and result sequences of π, respectively. The context C_0 is the initial state of π and the contexts $C_1, \ldots, C_{n-1}$ represent the influence of the environment on the computation.

If $C_i = \emptyset$ for all $i \geq 1$, then the reaction system is said to be working without the influence of the environment (or without environmental influence).

The sequence $sts(\pi) = W_0, \ldots, W_n$ denotes the state sequence of π, where $W_0 = C_0$ (the initial state), and $W_i = D_i \cup C_i$ for all $1 \leq i \leq n$.

The sequence $act(\pi) = E_0, \ldots, E_{n-1}$ of subsets of A such that $E_i = en(A, W_i)$ for all $0 \leq i \leq n-1$ represents the activity sequence of π.

Consequently, the evolution of the state sequence of $\mathcal{A}$ starting from W_0 is denoted by

$$W_0 \xrightarrow{E_0} W_1 \xrightarrow{E_1} \ldots \xrightarrow{E_{n-1}} W_n.$$

If $E_n = en(A, W_n) = \emptyset$, then the interactive process terminates.

Notice that the state sequence of a reaction system (working without the influence of the environment) is deterministic, each state has only one successor state. This is due to the property that reaction systems perform all enabled reactions on the given state.

2.2 Distributed reaction systems

A distributed reaction system consists of a finite set of reaction systems that either interact by input with a common environment or interact with each other by communicating products or reactions. The components are defined over a common background set and function as reaction systems on subsets of the background set assigned to them. These sets of objects are the current states of the components. The components of the distributed reaction system may communicate with each other. This can be realized by direct communication of products or reactions (cdcR(p) systems and cdcR(r) systems, see Csuhaj-Varjú et al. (2020)) or by sharing the products of the component with the products of its neighbours Bottoni et al. (2020). The latter variant, where the components are located in nodes of a virtual graph is called a network of reaction systems.

In this subsection, we recall a variant of distributed reaction systems where the contexts from the environment are generated by a context automaton, resembling to a nondeterministic finite automaton. This model is a variant of the so-called distributed reaction system and its operation, with slight modifications of the notions and notations originally introduced in Meski et al. (2019), and then in a modified form in Ciencialová L et al. (2022, 2023).

Definition 1 A *distributed reaction system* (a DRS for short) is a pair $\Delta = (S, \mathcal{A})$ where S is a finite nonempty set, the background set of Δ, and $\mathcal{A} = (A_1, \ldots, A_n)$ where A_i, $1 \leq i \leq n$, is a finite nonempty set of reactions over S. A_i is called the ith component of Δ, $1 \leq i \leq n$.

The distributed reaction system interacts with its environment. The environment is a finite set of reactants. The current reactant set in the environment may also be called the current context.

Now, we present the notion of a context automaton (or context provider) in Ciencialová L et al. (2022). This concept is a slightly modified variant of the notion of a context automaton in Meski et al. (2019).

Definition 2 Let $\Delta = (S, \mathcal{A})$ with $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$, be a distributed reaction system. A 5-tuple $M = (Q, C, R, q_0, F)$ is called a *context automaton* (or a context provider) for Δ if the following conditions are met:

- Q is a finite set, called the set of states of M,
- $C \subseteq \{(c_1, \ldots, c_n) \mid c_i \subseteq S, 1 \leq i \leq n\}$ is the finite set of n-tuples of contexts provided for components $A_1, \ldots, A_n$,
- $R \subseteq \{(q, (c_1, \ldots, c_n), r) \mid q, r \in Q, (c_1, \ldots, c_n) \in C\}$ is the set of transitions of M,
- $q_0 \in Q$ is the initial state of M, and
- $F \subseteq Q$ is the set of final states of M.

Notice that the context automaton is non-deterministic.

In the following, we extend the notion of a distributed reaction system to a distributed reaction system interacting with its environment, called an extended distributed reaction system. In the case of the extended distributed reaction system each component maintains its local state, which is a subset of S. A global state is an n-tuple of the local states of the components. The distributed reaction system is in interaction with its environment, i.e., at each transition from one global state to the next, the environment provides each component of the distributed reaction system with a context. The context is a finite, possibly empty set of elements of S.

Definition 3 An *extended distributed reaction system* (an EDRS for short) is a pair $\Gamma = (\Delta, M)$, where

- $\Delta = (S, \mathcal{A})$ with $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$, is a distributed reaction system over S, and
- $M = (Q, C, R, q_0, F)$ is a context automaton for Δ.

We now define the state of a distributed reaction system.

Definition 4 Let $\Gamma = (\Delta, M)$ be an extended distributed reaction system, where $\Delta = (S, \mathcal{A})$, $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$, $M = (Q, C, R, q_0, F)$.

A triplet $\sigma = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n))$ is called a *state* of Γ if $q \in Q$, $(c_1, \ldots, c_n) \in C$, and for $(d_1, \ldots, d_n)$ it holds that $d_i \subseteq S$, $1 \leq i \leq n$.

Each state $\sigma_0 = (q_0, (c_1, \ldots, c_n), (d_1, \ldots, d_n))$ where q_0 is the initial state of M is called an *initial state* of Γ, and each state $\sigma_f = (q_f, (c_1, \ldots, c_n), (d_1, \ldots, d_n))$ where $q_f \in F$ is called a *final state* of Γ.

Now we present the notion of a direct transition in an EDRS.

Definition 5 Let $\Gamma = (\Delta, M)$ be an extended distributed reaction system, where $\Delta = (S, \mathcal{A})$, $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$ and $M = (Q, C, R, q_0, F)$ is the context automaton for Γ.

Let σ_1 and σ_2 be two states of Γ, where $\sigma_1 = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n))$ and $\sigma_2 = (r, (c'_1, \ldots, c'_n), (d'_1, \ldots, d'_n))$. We say that there is a *direct transition* from σ_1 to σ_2 in Γ, denoted by $\sigma_1 \Longrightarrow \sigma_2$, if the following conditions are met:

- $(q, (c_1, \ldots, c_n), r) \in R$,
- $(c'_1, \ldots, c'_n) \in C$,
- $d'_i = res_{A_i}(c_i \cup d_i)$, $1 \leq i \leq n$.

The transitive reflexive closure of relation $\Longrightarrow$ is denoted by $\Longrightarrow^*$.

The non-deterministic nature of EDRSs can also be observed in the above definition. Notice that the only condition on $(c'_1, \ldots, c'_n)$ is that it be an element of C. So, a state of an EDRS may have more than one successor state.

Next, we define the notion of a finite interactive process in an extended distributed reaction system.

Definition 6 Let $\Gamma = (\Delta, M)$ be an extended distributed reaction system.

A finite sequence of states $\sigma_0, \ldots, \sigma_m$ of Γ is said to be a *finite interactive process* in Γ if $\sigma_0 \Longrightarrow \sigma_1 \Longrightarrow \ldots \Longrightarrow \sigma_{m-1} \Longrightarrow \sigma_m$, $m \geq 1$ holds for some initial state σ_0.

The finite interactive process is called *terminating* if σ_m is a final state of Γ. The set of all terminating finite interactive processes of Γ is denoted by Π_Γ.

Notice that if $\sigma_0 \Longrightarrow \sigma_1 \Longrightarrow \ldots \Longrightarrow \sigma_{m-1} \Longrightarrow \sigma_m$, $m \geq 1$, is a terminating interactive process, then $\sigma_0 \Longrightarrow \sigma_1 \Longrightarrow \ldots \Longrightarrow \sigma_{m-1} \Longrightarrow \sigma_m \Longrightarrow \sigma_{m+1} \Longrightarrow \ldots \Longrightarrow \sigma_{m+k}$,

$k \geq 1$, can also be a terminating interactive process if σ_{m+k} is a final state of Γ.

3 Distributed reaction systems with communication by request

In this section we introduce the concept of the distributed reaction system with communication by request. Unlike cdcR(p) systems, where the components send copies of the obtained products to certain target components, the components of these systems receive copies of products from certain other components upon request. A request, marked by a query (symbol), means that the issuing component requests the state of the target component. The target component sends the copies of the products in its state to the querying components provided that its state does not contain a query. This communication protocol also expresses hidden coordination of the function of the components. As mentioned above, the idea was inspired by parallel communicating grammar systems Păun and Kari (1989); Csuhaj-Varjú et al. (1994).

Definition 7 A distributed reaction system with *communication by request* (a qDRS for short) is an $(n+2)$-tuple $\Delta = (S, K, A_1, \ldots, A_n)$, $n \geq 1$, where

- S is a finite nonempty set, called the background set of Δ,
- $K = \{Q_1, \ldots, Q_n\}$ with $K \cap S = \emptyset$ is an alphabet called the set of queries of Δ where Q_i is associated to component A_i, $1 \leq i \leq n$.
- A_i, $1 \leq i \leq n$, is a finite nonempty set of extended reactions called the ith component of Δ. Each extended reaction is either a reaction over S, or a reaction over $S \cup K$ of the form $(R, I, \{q\})$, where R and I are nonempty disjoint subsets of S and $q \in K$ with $q \neq Q_i$.

If no confusion arises, we may use term reaction instead of extended reaction.

The extended reaction of the form $(R, I, \{q\})$ initiates the communication process. Notice that every such reaction introduces only one query and does not provide any product which is an element of the background set. Furthermore, self-query is not allowed.

A state of Δ is an n-tuple $\bar{D} = (D_1, \ldots, D_n)$, where $D_i \subseteq S \cup K$ and D_i is a non-empty set called the state of component A_i in $\bar{D}$. The set D_i is called query-free if $D_i \cap K = \emptyset$, $1 \leq i \leq n$. If every D_i, $1 \leq i \leq n$, is query-free, then $\bar{D}$ is called query-free.

Example 1 Let $\Delta = (\{a, b, c, X\}, \{Q_1, Q_2, Q_3\}, A_1, A_2, A_3)$ where

$$A_1 = \{(\{a\}, \{c\}, \{b\}),\ (\{b\}, \{c\}, \{a\})\},$$
$$A_2 = \{(\{b\}, \{X\}, \{a\}),\ (\{a\}, \{X\}, \{b\}),\ (\{b\}, \{X\}, \{Q_1\}),\ (\{b\}, \{X\}, \{Q_3\})\},$$
$$A_3 = \{(\{a\}, \{X\}, \{c\}),\ (\{c\}, \{X\}, \{a\}),\ (\{a\}, \{X\}, \{Q_1\}),\ (\{c\}, \{X\}, \{Q_1\})\}.$$

Then Δ is a qDRS with components A_1, A_2, A_3.

We define the (direct) transition between two states in Δ as follows.

Definition 8 Let $\Delta = (S, K, A_1, \ldots, A_n)$, $n \geq 1$, be a qDRS and let $\bar{D} = (D_1, \ldots, D_n)$ and $\bar{D}' = (D'_1, \ldots, D'_n)$ be two states of Δ.

We say that there is a (*direct*) *transition* between $\bar{D}$ and $\bar{D}'$ ($\bar{D}$ directly changes to $\bar{D}'$) if one of the following conditions holds:

- A transition of type (a), denoted by $\bar{D} \to \bar{D}'$, is performed if $D_i \cap K = \emptyset$ for each i, $1 \leq i \leq n$. In this case $D'_i = res_{A_i}(D_i)$.
- A transition of type (b), denoted by $\bar{D} \vdash \bar{D}'$, is performed if $D_k \cap K \neq \emptyset$ for some k, $1 \leq k \leq n$. In this case, for each i, $1 \leq i \leq n$, we write $D_i = X_i \cup K_i$, where $X_i = D_i \cap S$ and $K_i = \{Q_{i_1}, \ldots, Q_{i_r}\} = D_i \cap K$. Then $D'_i = X_i \cup K'_i \cup K''_i$ where $K'_i = \bigcup_{Q_{i_j} \in K_i, D_{i_j} \cap K = \emptyset} D_{i_j}$, and $K''_i = \bigcup_{Q_{i_j} \in K_i, D_{i_j} \cap K \neq \emptyset} \{Q_{i_j}\}$.

A transition of type (a) can also be called a reaction step and a transition of type (b) can be also be called a communication step of Δ. If the type of the transition is irrelevant, then we use notation $\bar{D} \Longrightarrow \bar{D}'$. In this case we use the short term transition.

We give a short explanation to the definition. If no query occurs in the state of any component, then the components perform all of their enabled reactions. The new state of the component will be the set of the obtained products. This is the case of transition of type (a).

If there is at least one component with a query, then transition of type (b) is performed. Suppose that a component with at least one query is A_i and its state is D_i. Then the new state D'_i of A_i consists of the union of the following sets: the set of reactants in D_i which are elements of S, the set of queries that request the state of such components which have non-query-free state, and the union of the states of the components with query-free state where a query to the component was issued. The states of the query-free components remain unchanged.

No transition of type (a) is possible if there exists a component with a state having a query as an element.

The transitive (and reflexive) closure of $\to$, $\vdash$, and $\Longrightarrow$ is denoted by $\to^+$, $\vdash^+$, and $\Longrightarrow^+$ ($\to^*$, $\vdash^*$, and $\Longrightarrow^*$) respectively.

A sequence of transitions is said to be a communication, if it is a sequence of communication steps. If the resulting state, σ_r, is query-free, then the communication is successful. Since Δ contains n components, the length of every successful communication is at most $n-1$.

Notice that a sequence of transitions resulting in a state $\bar{D_h}$ is halting in the following cases:

- The state $\bar{D_h}$ of Δ is query-free and no reaction step can be performed (i.e. there is no component with an enabled extended reaction), or
- $\bar{D_h}$ is not query-free and no communication step can be performed.

Note that if the transition sequence of a qDRS is halting, then it is finite. This follows from the fact that every qDRS has a finite state space, so any trajectory in it is either periodic (not halting) or finite (and halting).

A transition sequence is called terminating if it ends with a query-free state and is halting. A finite transition sequence of Δ starting from an initial state $\bar{D_0} = (D_{0,1}, \ldots, D_{0,n})$ is called a computation.

Definition 9 A sequence $\gamma = \bar{D_0}, \bar{D_1}, \ldots, \bar{D_m}, \ldots$ is called a *state sequence* of Δ if $\bar{D}_{i+1}$ can be obtained from $\bar{D_i}$ by a reaction step or by a communication step.

A subsequence $\gamma' = \bar{U_0}, \bar{U_1}, \ldots, \bar{U_r}, \ldots$ of the state sequence $\bar{D_0}, \bar{D_1}, \ldots, \bar{D_m}, \ldots$ is called a *state sequence in the strict sense* if $\bar{U_j}$ is a subset of S and there is no state $\bar{D_h}$ in γ' which is not an element of γ. Furthermore, if $\bar{U_j} = \bar{D_l}$ and $\bar{U}_{j+1} = \bar{D_s}$, then $l < s$ and there is no $\bar{D_r}$ where $l < r < s$ such that $\bar{D_r}$ is query-free.

If $\bar{D_0}, \bar{D_1}, \ldots, \bar{D_m}, \ldots$ is the sequence of k-tuples of the states of components $A_{i_1}, \ldots, A_{i_k}$ for some k, $1 \leq k \leq n$, then it is called the *state sequence of components* $A_{i_1}, \ldots, A_{i_k}$ of Δ.

Example 2 Let $(\{a\}, \{b\}, \{a\})$ be the initial state of $\Delta = (\{a, b, c, X\}, \{Q_1, Q_2, Q_3\}, A_1, A_2, A_3)$ from Example 1.

The following table demonstrates the first few transitions in Δ.

steps	state of A_1	state of A_2	state of A_3	transition type
0.	$\{a\}$	$\{b\}$	$\{a\}$	$\to$
1.	$\{b\}$	$\{a, Q_1, Q_3\}$	$\{c, Q_1\}$	$\vdash$
2.	$\{b\}$	$\{a, b, Q_3\}$	$\{c, b\}$	$\vdash$
3.	$\{b\}$	$\{a, b, c\}$	$\{c, b\}$	$\to$
4.	$\{a\}$	$\{a, b, Q_1, Q_3\}$	$\{a, Q_1\}$	$\vdash$
5.	$\{b\}$	$\{a, b, Q_3\}$	$\{a, b\}$	$\vdash$
6.	$\{b\}$	$\{a, b\}$	$\{a, b\}$	$\to$
7.	$\{a\}$	$\{a, b, Q_1, Q_3\}$	$\{c, Q_1\}$	

It can easily be seen that the states of the components are query-free after every third step and the length of each communication sequence is two.

Next, we show that every qDRS can be represented by a reaction system. More precisely, to the state sequence γ of a qDRS in the strict sense, starting from a given initial state, we can give a reaction system working without environmental influence and an initial state of this reaction system such that γ can be obtained as a mapping of a certain subsequence κ of the state sequence of the reaction system.

In order to avoid confusion, in the following theorem and in its proof, we use slightly different notations for reaction systems than usual, but their meaning is obvious from the context.

Theorem 1 *To any qDRS* $\Delta = (S, K, A_1, \ldots, A_n)$, $n \geq 1$, *with initial state* $\bar{D_0}$, *we can give a reaction system* $\mathcal{A} = (\Sigma, \mathcal{R})$, *where* Σ *is the background set of* $\mathcal{A}$ *and* $\mathcal{R}$ *is the set of reactions of* $\mathcal{A}$, $\mathcal{A}$ *works without environmental influence, an initial state* W_0 *of* $\mathcal{A}$, *and mappings* $h_i : \Sigma \to S \cup K$, $1 \leq i \leq n$, *such that the following holds*:

If $\bar{D_0}, \bar{D_1}, \ldots, \bar{D_i}, \ldots$ *is the state sequence of* Δ *in the strict sense, where* $\bar{D_i} = (D_{i,1}, \ldots D_{i,n})$, $i \geq 0$, $D_{i,l} \subseteq S$, $1 \leq l \leq n$, *and* $W_0, W_1 \ldots, W_r, \ldots$, $r \geq 0$ *is the state sequence of* $\mathcal{A}$, *then* $h_l(W_{i(n+1)}) = D_{i,l}$ *holds for each* $\bar{D_i}$, $1 \leq l \leq n$.

Proof The idea of the proof is the following. The reaction system $\mathcal{A}$ simulates each reaction step and each communication sequence in Δ in n steps. The elements of Σ represent the reactants and the queries of Δ together with their locations, i.e., with reference to the component where they are located. In addition, the current step number of the n-step simulation phase is indicated.

Let $\Sigma = S' \cup S'' \cup K' \cup \{X\}$ where S', S'', K' and X are defined as follows.

$$S' = \{[a, i] \mid a \in S, 1 \leq i \leq n\},$$
$$S'' = \{[a, i]^{(j)} \mid a \in S, 1 \leq i, j \leq n\},$$

where index i refers to the number of the component where a is located and j refers to the number of the step in the simulating transition sequence.

$$K' = \{[Q_k, i]^{(j)} \mid 1 \leq i, j, k \leq n, i \neq k\},$$

where reactant $[Q_k, i]^{(j)}$ in $\mathcal{A}$ represents a query of component A_i of Δ that requests the state of component A_k. Index j indicates that the simulation of the corresponding transition is in the jth step. In addition, we also have

$$X \in \Sigma \setminus (S' \cup S'' \cup K').$$

Since the maximal length of a communication in Δ is $n-1$, any reaction step and any communication in Δ will be simulated by n transitions in $\mathcal{A}$.

Reactions of $\mathcal{A}$ are given in the following manner.

If (R, I, P) is a (query-free) reaction in component A_i of Δ, we add the following reactions to $\mathcal{R}$:

$$(\{[a,i] \mid a \in R\}, \{[b,i] \mid b \in I\} \cup K', \{[c,i]^{(1)} \mid c \in P\}) \tag{1}$$

if $P \cap K = \emptyset$. Otherwise, if $P \cap K \neq \emptyset$, we have

$$(\{[a,i] \mid a \in R\}, \{[b,i] \mid b \in I\} \cup K', \\ \{[c,i]^{(1)} \mid c \in (P \cap S)\} \cup \{[Q_{k_1},i]^{(1)}, \ldots, [Q_{k_l},i]^{(1)}\}) \tag{2}$$

where $\{Q_{k_1}, \ldots, Q_{k_l}\} = P \cap K$ and $i \neq k_h$, $1 \leq h \leq l$.

We also add the following reactions to $\mathcal{R}$:

$$(\{[a,i]^{(j)}\}, \{X\}, [a,i]^{(j+1)}), \tag{3}$$

for $1 \leq j \leq n-1$, and

$$(\{[a,i]^{(n)}\}, \{X\}, \{[a,i]\}), \tag{4}$$

where $a \in S$.

The above reactions simulate reaction (R, I, P) of component A_i of Δ.

First, the enabled reactions are performed and the non-query products (certain elements of S) and possibly queries are indexed (see (1) and (2)). After that, a procedure consisting of n steps follows that preserves the non-query products while the communication takes place. During these n steps the non-query products (elements of S) are indexed by an increasing number until index n is obtained. After that the indexed version of the product is changed to the original one.

To simulate the communication steps, we add the following reactions to $\mathcal{R}$:

$$(\{[Q_k,i]^{(j)}, [a,k]^{(j)}\}, \cup_{l=1}^{n}\{[Q_l,k]^{(j)}\}, \{[a,i]^{(j+1)}\}), \tag{5}$$

for $a \in S$, $1 \leq j \leq n-1$, $1 \leq i,k \leq n$, where $i \neq k$.

These reactions represent the case when component A_i requests the actual state of component A_k provided that its state is query-free. Such a reaction checks whether or not the query exists and the state of A_k contains an element a of S (these are the reactants). It also checks whether or not there is a query in the state of A_k (the set of inhibitors), and then adds an a to the state of A_i. Notice if a is an element of the state of A_i, then this reaction implies no change in the state of A_i.

We also add the following reactions to $\mathcal{R}$:

$$(\{[Q_k,i]^{(j)}, [Q_l,k]^{(j)}\}, \{X\}, \{[Q_k,i]^{(j+1)}\}), \tag{6}$$

for $1 \leq j \leq n-1$ $1 \leq i,k,l \leq n$, where $i \neq k$ and $k \neq l$.

These reactions represent the case when component A_i requests the actual state of component A_k but the state of A_k is not query-free. These reactions check whether or not both A_i and A_k issue a query (the set of reactants), the inhibitor is irrelevant (singleton $\{X\}$), and then the query by A_i remains unsatisfied, thus the index of the corresponding symbol increases by 1 (the product). Notice that for a given query represented by $[Q_k,i]^{(j)}$ either reactions from the first set of reactions or reactions from the second set of reactions are enabled, but not reactions from both sets.

Now we show that the statement of the theorem holds. We first define mappings $h_i : \Sigma \to S \cup K$, $1 \leq i \leq n$.

Let $h_i([a,i]) = a$ for $[a,i] \in S'$, $h_i([a,i]^{(j)}) = a$ for $[a,i]^{(j)} \in S''$, and $h_i([Q_k,i]^{(j)}) = Q_k$, $1 \leq i,j,k \leq n$, $i \neq k$.

We extend h_i, $1 \leq i \leq n$, to subsets of Σ as follows. For every nonempty subset U of S' and for every nonempty subset V of S'', let $h_i(U) = \{h_i([a,i]) \mid [a,i] \in U\}$ and let $h_i(V) = \{h_i([a,i]^{(j)}) \mid [a,i]^{(j)} \in V\}$. For every nonempty subset Z of K' let $h_i(Z) = \{(h_i([Q_k,i]^{(j)}) \mid [Q_k,i]^{(j)} \in Z\}$, $1 \leq i,j,k \leq n$ $i \neq k$.

Finally, let $h_i(X) = X$, thus $h_i(\{X\}) = \{X\}$, and $h_i(\emptyset) = \emptyset$, for $1 \leq i \leq n$.

Let $\bar{D_0} = (D_{0,1}, \ldots, D_{0,n})$ be the initial state of Δ and let $W_0 = \{[a,i] \mid a \in D_{0,i}, 1 \leq i \leq n\}$ be the initial state of $\mathcal{A}$. It can immediately be seen that for every i, $h_i(W_0) = D_{0,i}$ holds.

Suppose that the statement holds for $\bar{D_j} = (D_{j,1}, \ldots, D_{j,n})$, $j \leq i$ and $D_{j,l} \subseteq S$, $1 \leq l \leq n$. (Note that $\bar{D_j}$ is an element of the state sequence of Δ in the strict sense.) That is, $h_l(W_{j(n+1)}) = D_{j,l}$, where $W_0, W_1 \ldots, W_r, \ldots$, $r \geq 1$ is the state sequence of $\mathcal{A}$. We show that for $\bar{D}_{i+1} = (D_{i+1,1}, \ldots, D_{i+1,n})$ it holds that $h_l(W_{(i+1)(n+1)}) = D_{i+1,l}$.

Let us consider the state $\bar{D_i} = (D_{i,1}, \ldots, D_{i,n})$. Since $h_l(W_{i(n+1)}) = D_{i,l}$, where $D_{i,l} \subseteq S$, we know that $W_{i(n+1)}$ consists of reactants of the form $[a, l]$, $1 \leq l \leq n$, where $a \in S$. The next state $\bar{D_i'} = (D'_{i,1}, \ldots, D'_{i,n})$ in the state sequence (not in the strict sense) of Δ will be either query-free or it will have at least one occurrence of a query. In the query-free case, starting from $W_{i(n+1)}$, computation steps are performed with reactions of type (1), then of type (3), (4). After performing the corresponding reactions of type (1), we obtain that $W_{i(n+1)+1}$ will consists of reactants of the form $[a,l]^{(1)}$, $1 \leq l \leq n$ and $W_{i(n+1)+1}$ corresponds to $\bar{D_i'} = (D'_{i,1}, \ldots, D'_{i,n})$. After this step, $n-1$ reaction steps follow (by using reactions of type (3)), where only the upper index of the reactants is increased by one until index n is obtained. Observe that these steps are codes of $W_{i(n+1)+1}$, thus they do not simulate any reaction or communication in Δ. In the next step, all reactants of the form $[a, l]^{(n)}$ are changed to $[a, l]$ (by reactions of type (4), thus the simulation of the reaction step is completed.

In the non query-free case, computation steps are performed on $W_{i(n+1)}$ with reactions of type (2), then of type (5), (6), and (3), (4) in this order. Suppose that

$W_{i(n+1)+1}$ corresponds to $\vec{D}_i' = (D_{i,1}', \ldots, D_{i,n}')$ in Δ, were at least one D_l', $1 \le l \le n$ contains a query. Then $W_{i(n+1)+1}$ consists of reactants of the form $[a,i]^{(1)}$ and $[Q_k,i]^{(1)}$ where $[a,i]^{(1)}$ represents reactant a at component A_i and $[Q_k,i]^{(1)}$ represents a query from component A_i to component A_k. By definition of the reactions of type (2), exactly the elements of $D_{i,l}'$, $1 \le l \le n$ are represented in $W_{i(n+1)+1}$. Suppose that the queries in $\vec{D}' = (D_{i,1}', \ldots, D_{i,n}')$ can be satisfied in l communication steps, where $1 \le l \le n$. Then, by using reactions of type (5) and (6) all communication steps in Δ are simulated. In each communication step, if $[Q_k,i]^{(j)}$ can be satisfied, i.e., component A_k does not contain a query, then $[Q_k,i]^{(j)}$ is replaced by the set of all reactants of the form $[a,k]^{(j+1)}$ that are present (reactions of type (5)), if $[Q_k,i]^{(j)}$ cannot be satisfied, i.e., the state of component A_k contains at least one query, then the upper index $[Q_k,i]^{(j)}$, j is increased by one (reactions of type 6). The upper index of the other reactants of the form $[a,i]^{(j)}$ is also increased by one (reactions of type (3)). Then, as before, either reactions of type (5) or reactions of type (6) are performed until no reactant of the form $[Q_k,i]^{(j)}$ is present. If the length of communication is less than $n-1$, then as in the previous case by reactions (3) and (4), the upper indices of the reactants are increased until n is obtained and after then all reactants are changed to be of the form $[a, i]$. In this way the communication is simulated. We can see that any computation in Δ is simulated by $\mathcal{A}$, and the state sequence of Δ in the strict sense can be obtained by mappings of a certain subsequence of the state sequence of $\mathcal{A}$. □

Example 3 We continue with the qDRS Δ presented in Example 1. Recall that $\Delta = (\{a,b,c,X\}, \{Q_1,Q_2,Q_3\}, (A_1,A_2,A_3)$ where

$$A_1 = \{(\{a\},\{c\},\{b\}),\ (\{b\},\{c\},\{a\})\},$$
$$A_2 = \{(\{b\},\{X\},\{a\}),\ (\{a\},\{X\},\{b\}),\ (\{b\},\{X\},\{Q_1\}),\ (\{b\},\{X\},\{Q_3\})\},$$
$$A_3 = \{(\{a\},\{X\},\{c\}),\ (\{c\},\{X\},\{a\}),\ (\{a\},\{X\},\{Q_1\}),\ (\{c\},\{X\},\{Q_1\})\}.$$

Let the initial state be $(\{a\},\{b\},\{a\})$.

Now we construct a reaction system $\mathcal{A} = (\Sigma, \mathcal{R})$ which simulates Δ. Let

$$S' = \{[a,i],[b,i],[c,i],[X,i] \mid 1 \le i \le 3\},$$

let

$$S'' = \{[a,i]^{(j)},[b,i]^{(j)},[c,i]^{(j)},[X,i]^{(j)} \mid 1 \le i,j \le 3\},$$

and consider symbol Y not in $S' \cup S''$. Let also

$$K' = \{[Q_k,i]^{(j)} \mid 1 \le i,j,k \le 3\},$$

and let the background set of $\mathcal{A}$ be

$$\Sigma = S' \cup S'' \cup K' \cup \{Y\}.$$

Consider the initial state to be $\{[a,1],[b,2],[a,3]\}$. In the following we provide the reactions in $\mathcal{A}$. For simplicity, we list only those which can be applied in the initial state.

Reaction

$$(\{[a,1]\},\{[c,1]\},\{[b,1]^{(1)}\})$$

of type (1) is in $\mathcal{R}$. Reactions

$$(\{[b,2]\},\{[X,2]\},\{[a,2]^{(1)}\}),\ (\{[b,2]\},\{[X,2]\},\{[Q_1,2]^{(1)}\}),$$
$$(\{[b,2]\},\{[X,2]\},\{[Q_3,2]^{(1)}\}),$$

and

$$([\{a,3]\},\{[X,3]\},\{[c,3]^{(1)}\}),\ ([\{a,3]\},\{[X,3]\},\{[Q_1,3]^{(1)}\})$$

are in $\mathcal{R}$ and of type (2).

For all $1 \le i \le 3$, $j = 1,2$, and $z \in \{a,b,c,X\}$, we have reactions

$$(\{[z,i]^{(j)}\},\{Y\},\{[z,i]^{(j+1)}\}),$$

of type (3), and reactions

$$(\{[z,i]^{(3)}\},\{Y\},\{[z,i]\}),$$

of type (4) in $\mathcal{R}$.

For simplicity, in the following we provide only those reactions which are enabled during the computation. Reactions

$$(\{[Q_1,2]^{(1)},[b,1]^{(1)}\},\{[Q_1,1]^{(1)},[Q_2,1]^{(1)},[Q_3,1]^{(1)}\},\{[b,2]^{(2)}\}),$$

$$(\{[(Q_1,3]^{(1)},[b,1]^{(1)}\},\{[Q_1,1]^{(1)},[Q_2,1]^{(1)},[Q_3,1]^{(1)}\},\{[b,3]^{(2)}\}),$$
$$(\{([Q_3,2]^{(2)},[b,3]^{(2)}\},\{[Q_1,3]^{(2)},[Q_2,3]^{(2)},[Q_3,3]^{(2)}\},\{[b,2]^{(3)}\}),$$
$$(\{([Q_3,2]^{(2)},[c,3]^{(2)}\},\{[Q_1,3]^{(2)},[Q_2,3]^{(2)},[Q_3,3]^{(2)}\},\{[c,2]^{(3)}\}),$$

are reactions of type (5) in $\mathcal{R}$. Finally, let

$$(\{([Q_3,2]^{(1)},[Q_1,3]^{(1)}\},\{Y\},\{[Q_3,2]^{(2)}\}),$$
$$(\{([Q_3,2]^{(1)},[Q_2,3]^{(1)}\},\{Y\},\{[Q_3,2]^{(2)}\}),$$
$$(\{([Q_3,2]^{(1)},[Q_3,3]^{(1)}\},\{Y\},\{[Q_3,2]^{(2)}\}),$$

be reactions of type (6) in $\mathcal{R}$.

Then the state sequence of $\mathcal{A}$ is the following:

step	state	computation
0.	$\{[a,1],[b,2],[a,3]\}$	$\Longrightarrow$
1.	$\{[b,1]^{(1)},[a,2]^{(1)},[Q_1,2]^{(1)},[Q_3,2]^{(1)},[c,3]^{(1)},[Q_1,3]^{(1)}\}$	$\Longrightarrow$
2.	$\{[b,1]^{(2)},[a,2]^{(2)},[b,2]^{(2)},[Q_3,2]^{(2)},[c,3]^{(2)},[b,3]^{(2)}\}$	$\Longrightarrow$
3.	$\{[b,1]^{(3)},[a,2]^{(3)},[b,2]^{(3)},[c,2]^{(3)},[c,3]^{(3)},[b,3]^{(3)}\}$	$\Longrightarrow$
4.	$\{[b,1],[a,2],[b,2],[c,2],[c,3],[b,3]\}$	

In the first computation step all of the reactants $[a, 1]$,$[b, 2]$, $[a, 3]$ get the upper index (1) and the corresponding reactions of A_i, $1 \leq i \leq 3$ are simulated. This is done by applying reactions of type (1) and (2) in $\mathcal{A}$. In the second step, the upper index of the reactants $[b,1]^{(1)}$, $[a,2]^{(1)}$, $[c,3]^{(1)}$ increases to (2) and in the meantime the two reactants $[Q_1,2]^{(1)}$ and $[Q_1,3]^{(1)}$, representing queries Q_1 in Δ, are changed for $[b,2]^{(2)}$ and $[b,3]^{(2)}$, respectively, simulating the corresponding communication step in Δ. This step is done by performing reactions of type (3) and (5) of $\mathcal{A}$. Reactant $[Q_3,2]^{(1)}$ is changed for $[Q_3,2]^{(2)}$ by a reaction of type (6) of $\mathcal{A}$, since this query in the simulated transition step in Δ cannot be satisfied. In the third step, $[Q_3,2]^{(2)}$ is replaced by $\{[c,3]^{(3)},[b,3]^{(3)}\}$, according to reactions of type (5), and the upper index of the other reactants is increased by one. In the fourth step, the reactants are changed to their original variant without upper index. During this procedure, no more reactions of $\mathcal{A}$ can be performed. Hence the transitions in $\mathcal{A}$ simulate the transitions in Δ. Furthermore, if we define $h_i([x,i]) = x$, for $x \in \{a,b,c\}$, $1 \leq i \leq 3$, then we obtain the statement of the result.

Definition 10 Let $\Gamma = (\Delta, M)$ be an extended distributed reaction system, and let $\sigma_0, \ldots, \sigma_m, \ldots$ be a state sequence in Γ, where each state σ_i is of the form $(q_i,(C_{i,1},\ldots,C_{i,n}),(D_{i,1},\ldots,D_{i,n}))$, $i \geq 1$.

The sequence $(D_{i,1},\ldots,D_{i,n})$, $i \geq 0$ is called the *n-tuple of state sequences of the components* of Γ starting from the initial state $\sigma_0 = (D_{0,1},\ldots,D_{0,n})$. If no confusion arises, we use the wording the *state sequence of the components* of Γ.

We show that to any qDRS with an initial state we can give an EDRS with an initial state such that the state sequence of the qDRS and the state sequence of the components of the EDRS coincide.

Theorem 2 *To any qDRS* $\Delta = (S,K,A_1,\ldots,A_n)$, $n \geq 1$ *and initial state* $\bar{D}_0$, *we can give an EDRS* $\Gamma = (\Delta', M)$ *with initial state* σ_0 *such that the state sequence of* Δ *and the state sequence of the components of* Γ *coincide.*

Proof Let Γ have the following components: $\Delta' = (\Sigma, \mathcal{A}')$ where $\mathcal{A}' = (A'_1,\ldots,A'_n)$, $n \geq 1$, is the distributed reaction system of Γ and $M = (Q,C,R,q_0,F)$ is the context automaton.

Before providing the components of Γ, we discuss the computation in Δ. It is easy to see that for any state $(D_1,\ldots,D_n)$ of Δ, we can determine the only state $(D'_1,\ldots,D'_n)$ of Δ where $(D_1,\ldots,D_n) \Longrightarrow (D'_1,\ldots,D'_n)$ holds. Our construction will be based on this observation.

We define the components of M together with the reaction sets A'_i, $1 \leq i \leq n$, of Δ' of Γ.

For every state $(D_1,\ldots,D_n)$ of qDRS Δ, $[D_1,\ldots,D_n]$ is a state of M and M has no more elements. The initial state of M is $[D_{0,1},\ldots,D_{0,n}]$, where $\bar{D}_0 = (D_{0,1},\ldots,D_{0,n})$ is the initial state of Δ. The set of final states F of M consists of those elements $[D_1,\ldots,D_n]$ of Q where $(D_1,\ldots,D_n)$ is a query-free state of Δ and there is no state $(D'_1,\ldots,D'_n)$ of Δ where $(D_1,\ldots,D_n) \Longrightarrow (D'_1,\ldots,D'_n)$ holds.

The background set Σ is the union of the following disjoint sets:

$$\Sigma = S \cup K \cup \{X_{[D_1,\ldots,D_n]} \mid (D_1,\ldots,D_n) \text{ is a state of } \Delta\} \cup \{Z\}$$

where Z is an auxiliary element, different from all other elements of Σ.

Now, we define the elements of R, the set of transitions of M.

If $(D_1,\ldots,D_n) \Longrightarrow (D'_1,\ldots,D'_n)$ is a transition in Δ then we add

$$([D_1,\ldots,D_n],(X_{[D_1,\ldots,D_n]},\ldots,X_{[D_1,\ldots,D_n]}),[D'_1,\ldots,D'_n])$$

to R. Notice that $X_{[D_1,\ldots,D_n]}$ is not in $S \cup K$; $\{X_{[D_1,\ldots,D_n]}\}$ is the context added to the components.

Now we define the reactions of components of $\mathcal{A}' = (A'_1,\ldots,A'_n)$ of Γ.

(1) If $(D_1,\ldots,D_n) \Longrightarrow (D'_1,\ldots,D'_n)$ is a transition in Δ and $(D_1,\ldots,D_n)$ is query-free, then we add the reaction

$$(\{X_{[D_1,\ldots,D_n]}\} \cup D_i, S \setminus D_i, D'_i)$$

to A'_i, $1 \leq i \leq n$. Since $D'_i = res_{A_i}(D_i)$, the above reaction simulates the transition in Δ. Notice that symbol $X_{[D_1,\ldots,D_n]}$ indicates which transition is to be performed, thus transitions cannot be mixed.

(2) If $(D_1,\ldots,D_n) \vdash (D'_1,\ldots,D'_n)$ is a transition in Δ where $(D_1,\ldots,D_n)$ is not query-free, then we add the following reactions to A'_i. We add

$$(\{X_{[D_1,\ldots,D_n]}\} \cup (D_i \cap S), S \setminus D_i, D_i \cap S)$$

since those elements which are reactants in Δ need to remain unchanged.

For all k such that $D_k \cap K = \emptyset$, we also add

$$(\{X_{[D_1,\ldots,D_n]}, Q_k\}, \{Z\}, D_k).$$

If component i asked for the state of component k (and D_k is query-free), then D_k is added to the state of component i and the query disappears. Notice that $X_{[D_1,\ldots,D_n]}$ plays a crucial role in this reaction and that we can decide whether or not D_k contains symbol representing a query.

For the case when D_k contains at least one query, we add

$$(\{X_{[D_1,\ldots,D_n]}, Q_k\}, \{Z\}, \{Q_k\}).$$

As in the previous case, the role of $X_{[D_1,\ldots,D_n]}$ is significant.

Now we prove that the state sequence of Δ and the state sequence of the components of Γ coincide.

If the initial state of Γ is

$$([D_{0,1},\ldots,D_{0,n}],(X_{[D_{0,1},\ldots,D_{0,n}]},\ldots,X_{[D_{0,1},\ldots,D_{0,n}]}),(D_{0,1},\ldots,D_{0,n})),$$

then by definition, the initial state of Δ is equal to the initial state of (the components of) Γ. Assume that the equality of the two state sequences (the statement of the theorem) holds for up to the ith pair of states, where $i \geq 1$. We show that the statement also holds for the $(i+1)$th pair.

Suppose that the ith element of the state sequence κ in Δ is $(D_1,\ldots,D_n)$ and $(D_1,\ldots,D_n)$ is also the ith element of the state sequence of Γ. Then either there exists a unique state $(D'_1,\ldots,D'_n)$ in Δ where $(D_1,\ldots,D_n) \Longrightarrow (D'_1,\ldots,D'_n)$ holds, or the computation cannot be continued.

If $(D_1,\ldots,D_n)$ is query-free, then for $(D'_1,\ldots,D'_n)$ it holds that $D'_j = res_{A_j}(D_j)$. Let us consider now the case where $(D_1,\ldots,D_n)$ is the ith element of the state sequence of Γ.

In this case, there exists a transition

$$[D_1,\ldots,D_n],(X_{[D_1,\ldots,D_n]},\ldots,X_{[D_1,\ldots,D_n]}),[D''_1,\ldots,D''_n])$$

in the automaton M. Then, by the construction of M, see above, $D'_j = D''_j$ holds for $1 \leq j \leq n$.

Let us suppose now that $(D_1,\ldots,D_n)$, the ith element of κ is not query-free, i.e., there exists at least one component A_l such that D_l contains a query. By definition, for every such D_l, reaction $(\{X_{[D_1,\ldots,D_n]}\} \cup (D_l \cap S), S \backslash D_l, D_l \cap S)$ is applied, that is, those elements which are reactants in Δ remain unchanged. Furthermore, reaction $(\{X_{[D_1,\ldots,D_n]}, Q_k\}, \{Z\}, D_k)$ for $D_k \cap K \neq \emptyset$ is also applied, if possible. This means that component A_i asked for the state of component A_k provided that D_k is query-free. In this case D_k is added to the state of component D_l and the reactant representing the query disappears from the state. If D_k contains at least one query, then $(\{X_{[D_1,\ldots,D_n]}, Q_k\}, \{Z\}, \{Q_k\})$ is applied, i.e., Q_k remains unchanged.

The procedure is repeated until the new state will be free from reactants representing a query. Notice the synchronized behaviour of the components. Since the performed reactions simulated the communication, we obtain that the statement holds for the $(i+1)$th pair of the two sequences as well. □

Example 4 We start from the qDRS given in Example 1. Recall that $\Delta = (\{a,b,c,X\},\{Q_1,Q_2,Q_3\},(A_1,A_2,A_3)$ has components

$$A_1 = \{(\{a\},\{c\},\{b\}),\ (\{b\},\{c\},\{a\}),$$
$$A_2 = \{(\{b\},\{X\},\{a\}),\ (\{a\},\{X\},\{b\}),\ (\{b\},\{X\},\{Q_1\}),\ (\{b\},\{X\},\{Q_3\})\},$$
$$A_3 = \{(\{a\},\{X\},\ \{c\}),(\{c\},\{X\},\{a\}),\ (\{a\},\{X\},\{Q_1\}),\ (\{c\},\{X\},\{Q_1\})\}.$$

Let the initial state of $\Delta =$ be $(\{a\},\{b\},\{a\})$. To help the easier reading, we recall the first few transitions of Δ.

steps	state of A_1	state of A_2	state of A_3	transition type
0.	$\{a\}$	$\{b\}$	$\{a\}$	$\rightarrow$
1.	$\{b\}$	$\{a,Q_1,Q_3\}$	$\{c,Q_1\}$	$\vdash$
2.	$\{b\}$	$\{a,b,Q_3\}$	$\{c,b\}$	$\vdash$
3.	$\{b\}$	$\{a,b,c\}$	$\{c,b\}$	

Now we construct the simulating EDRS Γ For simplicity, we present only those elements of $\Gamma = (\Delta', M)$ which are necessary to understand the main ideas of the simulation.

according to the proof of Theorem 2,

$$s_0 = [(\{a\},\{b\},\{a\}],$$
$$s_1 = [\{b\},\{a,Q_1,Q_3\},\{c,Q_1\}]$$
$$s_2 = [\{b\},\{a,b,Q_3\},\{c,b\}],$$
$$s_3 = [\{b\},\{a,b,c\},\{c,b\}]$$

are states of M and $s_0 = [(\{a\},\{b\},\{a\}]$ is its initial state.

By definition, the following transitions are in the transition set of M:

The transition

$$([(\{a\},\{b\},\{a\}],(\{X_{s_0}\},\{X_{s_0}\},\{X_{s_0}\}),[\{b\},\{a,Q_1,Q_3\},\{c,Q_1\}])$$

in M provides component A'_i, $1 \leq i \leq 3$, with the same context, $\{X_{s_0}\}$, and then M enters in state $s_1 = [\{b\},\{a,Q_1,Q_3\},\{c,Q_1\}]$.

Similarly,

$$([\{b\},\{a,Q_1,Q_3\},\{c,Q_1\}],(\{X_{s_1}\},\{X_{s_1}\},\{X_{s_1}\}),[\{b\},$$
$$\{a,b,Q_3\},\{c,b\}])$$

is a transition in M which provides the same context, $\{X_{s_1}\}$, for components A'_1, A'_2, A'_3, respectively.

Finally,

$$([\{b\},\{a,b,Q_3\},\{c,b\}],(\{X_{s_2}\},\{X_{s_2}\},\{X_{s_2}\}),[\{b\},$$
$$\{a,b,c\},\{c,b\}])$$

is a transition in M which provides the same context, $\{X_{s_2}\}$, for components A'_1, A'_2, A'_3, respectively.

We present the corresponding reactions of components A'_i, $1 \leq i \leq 3$. For simplicity, only those reactions are provided that we will used in the sequel.

Reactions

$$\rho_{1,1}:(\{a,X_{s_0}\},\{b,c\},\{b\}),\ \rho_{2,1}:(\{b,X_{s_1}\},\{a,c\},\{b\}\}),$$
$$\rho_{3,1}:(\{b,X_{s_2}\},\{a,c\},\{b\})$$

are in component A'_1, and reactions

$$\rho_{1,2}:(\{b,X_{s_0}\},\{a,c\},\{a,Q_1,Q_3\}),\ \rho_{2,2}:(\{a,X_{s_1}\},\{b,c\},\{a\}),$$
$$\rho_{3,2}:(\{Q_1,X_{s_1}\},\{b,c\},\{b\}),\ \rho_{4,2}:(\{Q_3,X_{s_1}\},\{b,c\},\{Q_3\}),$$
$$\rho_{5,2}:(\{a,b,X_{s_2}\},\{c\},\{a,b\}),\ \rho_{6,2}:(\{Q_3,X_{s_2}\},\{c\},\{b,c\})$$

are in component A'_2.

Finally, reactions

$$\rho_{1,3} : (\{a, X_{s_0}\}, \{b, c\}, \{c, Q_1\}),\ \rho_{2,3} : (\{c, X_{s_1}\}, \{a, b\}, \{c\}),$$
$$\rho_{3,3} : (\{Q_1, X_{s_1}\}, \{a, b\}, \{b\}),\ \rho_{4,3} : (\{b, c, X_{s_2}\}, \{a\}, \{b, c\})$$

are in A'_3.

We now present the first few computation steps in Γ, see below.

steps	state of Γ	reactions applied
0.	$(s_0, (\{X_{s_0}\}, \{X_{s_0}\}, \{X_{s_0}\}), (\{a\}, \{b\}, \{a\}))$	$\rho_{1,1}, \rho_{1,2}, \rho_{1,3}$
1.	$(s_1, (\{X_{s_1}\}, \{X_{s_1}\}, \{X_{s_1}\}), (\{b\}, \{a, Q_1, Q_3\}, \{c, Q_1\}))$	$\rho_{2,1}, \rho_{2,2}, \rho_{3,2}, \rho_{4,2}, \rho_{2,3}, \rho_{3,3}$
2.	$(s_2, (\{X_{s_2}\}, \{X_{s_2}\}, \{X_{s_2}\}), (\{b\}, \{a, b, Q_3\}, \{c, b\}))$	$\rho_{3,1}, \rho_{5,2}, \rho_{6,2}, \rho_{4,3}$
3.	$(s_3, (\{X_{s_3}\}, \{X_{s_3}\}, \{X_{s_3}\}), (\{b\}, \{a, b, c\}, \{c, b\}))$	

The initial state of the components is $(\{a\}, \{b\}, \{c\})$, as it was mentioned above. It can be seen that the state sequence of the components of Γ, starting with their initial state, is equal to the state sequence of Δ.

Notice that the components of both EDRS and qDRS receive contexts (sets of reactants) from outside. The components of the EDRS obtain context from the context automaton, and the components of the qDRS obtain contexts by queries from other components. Since the context automaton can be non-deterministic, it is possible that different context sets are added to the components of an EDRS during transitions even when they start in the same configuration. In qDRS, however, this cannot happen, as the components are deterministic in the sense that the results obtained by any transition is completely determined by the starting configuration. Furthermore, in the case of the qDRS, the components are allowed to communicate with more than one component, while the components of an EDRS are in communication only with the context-automaton. In order to construct a qDRS which simulates an EDRS, we should resolve this difference.

Although we do not have a solution for the general case, we show a connection between star-type qDRS and a special simple variant of EDRSs.

A qDRS Δ is called *star-type* if there is only one component to which a query can be issued.

Definition 11 Let $\Gamma = (\Delta, M)$ be an extended distributed reaction system where $\Delta = (S, \mathcal{A})$, $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$ and $M = (Q, C, R, q_0, F)$. The system Γ is called *very simple* if for every state $q \in Q$, the context automaton M has at most one transition $(q, (c_1, \ldots, c_n), r) \in R$.

Next we show that if an EDRS with a given initial state is very simple (in the sense defined above), then we can give a qDRS and an initial state such that the state sequence of the qDRS and the state sequence of the EDRS correspond to each other.

Theorem 3 *Let $\Gamma = (\Delta, M)$ with $\Delta = (S, \mathcal{A})$, $\mathcal{A} = (A_1, \ldots, A_n)$ be a very simple EDRS, and let σ_0 be one of its initial states. Then we can give a star-type qDRS $\Delta' = (S', K', A'_0, A'_1, \ldots, A'_n)$ with $n+1$ components and initial state $\bar{D}_0$, such that every fifth state in the state sequence of the components $A'_1, \ldots, A'_n$ of Δ' and the state sequence of the components $(A_1, \ldots, A_n)$ of Γ coincide.*

Proof Let $\Gamma = (\Delta, M)$ be an extended distributed reaction system where $\Delta = (S, \mathcal{A})$, $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$, the context automaton is $M = (Q, C, R, q_0, F)$, and let $\sigma_0 = (q_0, (C_{0,1}, \ldots, C_{0,n}), (D_{0,1}, \ldots, D_{0,n}))$ be the initial state of Γ.

Let

$$R = \{tr_q \mid tr_q = (q, (C_1 \ldots, C_n), r) \in R \text{ for some } q, r \in Q\}$$

be the set of transitions in M. Notice that the initial state of M is q_0, and there is a finite set $F \subseteq Q$ of final states of M. Note also, that since Γ is very simple, each transition in Γ either has no successor transition or it has only exactly one successor transition, so we can represent each transition in R by its starting state.

Now we give $\Delta' = (S', K', A'_0, A'_1, \ldots, A'_n)$, and define its initial state as

$$\bar{D}_0 = (\{tr_{q_0}\}, D_{0,1}, \ldots, D_{0,n}).$$

Notice that no query symbol appears in $\bar{D}_0$.

The basic idea of the proof is the following: Component A'_0 represents the transitions of the context automaton. To simulate a transition in Γ, components $A'_1, \ldots, A'_n$ request the state of A'_0 and add the context provided for them to their current state. Then they perform the enabled reactions.

We define the components of Δ' as follows.

Let $S' = S \cup S'' \cup S_R \cup \{X\}$, where

$$S'' = \{a', a'' \mid a \in S\},$$
$$S_R = \{[tr_q], [tr_q]', [tr_q]'' \mid tr_q \in R,\ q \in Q\}.$$

Note that the short notation tr_q in the symbols $[tr_q], [tr_q]'$, and $[tr_q]''$ stands for the unique transition $(q, (C_{q,1} \ldots, C_{q,n}), r) \in R$.

Component A'_0 contains, for each $tr_q = (q, (C_{q,1} \ldots, C_{q,n}), r) \in R$, the following reactions:

(a) $(\{[tr_q]\}, \{X\}, \{[tr_q]'\})$,
(b) $\{[tr_q]'\}, \{X\}, \{[tr_q]''\})$,
(c) $\{[tr_q]''\}, \{X\}, \{[tr_r]\})$,

where $tr_r = (r, (C'_1 \ldots, C'_n), s) \in R$ is also a transition in M.

We define the reactions of the other components. Let each A'_i, $1 \leq i \leq n$, have the following reactions:

$$\begin{array}{ll}(1) & \{(\{a\},\{X\},\{a'\}) \mid a \in S\} \cup \{(\{a\},\{X\},\{Q_0\})\} \mid a \in S\},\\ (2) & \{(\{a'\},\{X\},\{a''\}) \mid a \in S\} \cup \{([tr_q]'\},\{X\},C''_{q,i}) \mid tr_q \in R\},\\ (3) & \{U'',I,P) \mid (U,I,P) \in A_i\},\end{array}$$

where $U'' = \{a'' \mid a \in U\}$ and $C''_{q,i} = \{a'' \mid a \in C_{q,i}\}$ for $tr_q = (q,(C_{q,1},\ldots,C_{q,n}),s)$.

Each component A'_i, $1 \leq i \leq n$ works with the following phases of steps:

Let us consider a state $\bar{D}$ of Δ' where the state D_i of component A'_i is a subset of S. The component A_0 has only one reactant as its state, say, $[tr_q]$, that represents the transition in M to be performed. Then, component A'_i changes each reactant a in its current state to its primed version, a', and at the same time, by a query Q_0 asks for the state of component A'_0. This is performed by reactions of type (1). No more reactions can be performed in this state.

During the same step, component A'_0 changes $[tr_q]$ to $[tr_q]'$, by reaction of type (a). Thus, in the next state of Δ', component A'_i, $1 \leq i \leq n$, has the primed version of its reactants from S together with the product Q_0, while component A'_0 has $[tr_q]'$.

In the following step, communication takes place, and after that in the new state of Δ, components A'_i will have as state the same elements as before, except that Q_0 is changed or $[tr_q]'$. The state of component A'_0 remains unchanged, $[tr_q]'$.

Then the state of component A'_i, $1 \leq i \leq n$ changes as follows: $[tr_q]'$ is changed to $C''_{q,i}$, the double-primed version of the context provided for component A_i by the transition $tr_q = (q,(C_{q,1}\ldots,C_{q,n}),r)$ in M represented by $[tr_q]$. At the same time, all reactants a' change to a''. This is performed by reactions of type (2). The state of component A'_0, $[tr]'$ is changed to $[tr]''$ by a reaction of type (b).

Finally, all enabled reactions of the form (U'',I,P) of A'_i, $1 \leq i \leq n$, are performed, where (U, I, P) is a reaction in A_i. This is done by reactions of type (3). In the meantime, $[tr_q]''$ at A'_0 is changed to $[tr_r]$ by reaction of type (c). Notice that $[tr_r]$ corresponds to the new transition to be performed, $t_r = (r,(C_{r,1},\ldots,C_{r,n}),s)$.

As the result, we obtain that the n-tuple of states of components $A'_1,\ldots,A'_n$ is equal to the state of the components of Γ obtained from the state $(D_1,\ldots,D_n)$ after the transition tr_q is performed in M. Notice that due to the fact that Γ with σ_0 is very simple, each transition in M has only one successor transition. This property is used in the simulation.

We leave the further details of the proof to the reader. □

We present a simple example to demonstrate how the construction in the proof works.

Example 5 Let $\Gamma = (\Delta, M)$ be an EDRS, and suppose that the context automaton M has two transitions $tr_{q_1} = (q_1,(\{b\},\{b\}),q_2)$ and $tr_{q_2} = (q_2,(\{a\},\{b\}),q_3)$. Let M be in state q_1, let component A_1 be in state $\{b\}$, and component A_2 be in state $\{a\}$.

Let us suppose that A_1 has reactions $(\{a\},\{X\},\{b\})$, $(\{b\},\{X\},\{a\})$ and A_2 has reactions $(\{b\},\{X\},\{a\})$, $(\{a\},\{X\},\{a\})$. Then M enters state q_2, component A_1 enters $\{a,b\}$, and A_2 stays in state $\{a\}$.

We construct the components A'_0, A'_1, and A'_2 of the simulating qDRS Δ' as follows. A'_0 contains the following reactions:

$$(\{[tr_{q_1}]\},\{X\},\{[tr_{q_1}]'\}),(\{[tr_{q_1}]'\},\{X\},\{[tr_{q_1}]''\}),\\ (\{[tr_{q_1}]''\},\{X\},\{[tr_{q_2}]\}).$$

Component A'_1 has the following reactions:

$$(\{a\},\{X\},\{a'\}),(\{a\},\{X\},\{Q_0\}),(\{a'\},\{X\},\{a''\}),\\ (\{a''\},\{X\},\{b\}),\\ (\{[tr_{q_1}]''\},\{X\},\{b\}).$$

Component A'_2 has the following reactions:

$$(\{b\},\{X\},\{b'\}),(\{b\},\{X\},\{Q_0\}),(\{b'\},\{X\},\{b''\}),\\ (\{b''\},\{X\},\{a\}),\\ (\{[tr_{q_1}]''\},\{X\},\{b\}).$$

The computation in Δ' is as follows:

state of A'_0	state of A'_1	state of A'_2	transition type
$\{[tr_{q_1}]\}$	$\{b\}$	$\{a\}$	$\rightarrow$
$\{[tr_{q_1}]'\}$	$\{a',Q_0\}$	$\{b',Q_0\}$	$\vdash$
$\{[tr_{q_1}]'\}$	$\{a',[tr_{q_1}]'\}$	$\{b',[tr_{q_1}]'\}$	$\rightarrow$
$\{[tr_{q_1}]''\}$	$\{a'',b''\}$	$\{b''\}$	$\rightarrow$
$\{[tr_{q_2}]\}$	$\{a,b\}$	$\{a\}$	

As we can see that the states correspond to each other.

4 Distributed reaction systems and multihead finite automata

In this section we relate languages of distributed reaction systems to languages of multihead finite automata.

A (nondeterministic) *one-way k-head finite automaton*, a 1NFA(k) in short, is a construct $M = (Q,\Sigma,k,\delta,\$,q0,F)$ where Q is the finite set of states, Σ is the set of input symbols, $k \geq 1$ is the number of heads, $S \notin \Sigma$ is the end-marker, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of accepting states, and partial function $\delta : Q \times (\Sigma \cup \{\lambda,\$\})^k \rightarrow Q$ is the transition function. Whenever $q' \in \delta(q,x_1,\ldots,x_k)$ is defined, the state q can be changed to q' if the input contains $x_i \in \Sigma \cup \{\$\}$ at the position of the i-th reading head for $x_i \neq \lambda$, or if $x_i = \lambda$, then the input might contain an arbitrary symbol from $\Sigma \cup \{\$\}$, $1 \leq i \leq k$. It is also assumed that in the case of $x_i \neq \lambda$, the reading head moves one position to the right during the transition, but it does not move and stays its current position if $x_i = \lambda$, $1 \leq i \leq k$.

The configuration of a 1NFA(k) can be denoted by $(q, w_1\$, \ldots, w_n\$)$ where $q \in Q$ and w_i, $1 \leq i \leq k$, are the parts of the input string which are not yet read by the corresponding heads.

Remark 1 The customary way of presenting the transition functions of multihead finite automata with input alphabet Σ is by $\delta : Q \times (\Sigma)^k \to Q \times (\{0,1\})^k$ where for $(q', d_1, \ldots, d_k) \in \delta(q, a_1, \ldots, a_k)$, $q' \in Q$ is the new state, the symbols $a_i \in \Sigma$, $1 \leq i \leq k$ are at the positions of the reading heads in the input, and $(d_1, \ldots, d_k) \in (\{0,1\})^k$ denotes the movements of the heads: if for some j, $d_j = 1$, then the j-th head is moved one cell to the right during the transition, but it stays at its current position if $d_j = 0$, $1 \leq j \leq k$. It is not difficult to see that our definition given above is equivalent to the customary.

We will also assume, without the loss of generality, that 1NFA(k) accept after each head has scanned the entire tape, that is, by entering the final state only after each head has read (and moved to the right of) the endmarker symbol \$.

In the following, we consider two possible ways of describing languages with extended distributed reaction systems. The idea is to assign symbols of an alphabet to the states of the components of the system, and to obtain this way sequences of symbols (or words) by terminating interactive processes corresponding to each of the components. The seminal variant of the concept was introduced in Ciencialová L et al. (2022).

Definition 12 Let $\Gamma = (\Delta, M)$ be an EDRS where $\Delta = (S, \mathcal{A})$ with $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$, is a distributed reaction system over S, $M = (Q, C, R, q_0, F)$ is the context automaton for Γ. Let Q_Γ be the set of all states of Γ. We define mappings ρ, ϕ, ϕ_i as follows.

1. Let Σ be an alphabet such that $card(\Sigma) = card(2^S) - 1$. We define a bijective mapping $\rho : 2^S \to \Sigma \cup \{\lambda\}$ such that $\rho(d) = \lambda$ if and only if $d = \emptyset$.
2. Mapping $\phi : Q_\Gamma \to \underbrace{\Sigma^* \times \ldots \times \Sigma^*}_{n\ times}$ is defined as $\phi(\sigma) = (\rho(d_1), \ldots, \rho(d_n))$ for each state $\sigma = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n)) \in Q_\Gamma$.
3. Mapping $\phi_i : Q_\Gamma \to \Sigma^*$, $1 \leq i \leq n$, is defined as $\phi_i(\sigma) = \rho(d_i)$ for each state $\sigma = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n)) \in Q_\Gamma$.

Notice that the mapping ρ assigns to any nonempty subset of S a letter of Σ. Note also that ρ is a bijective mapping, thus, there is no letter in Σ that is the map of two different nonempty subsets of S.

Definition 13 Let $\Gamma = (\Delta, M)$ be an EDRS where $\Delta = (S, \mathcal{A})$ with $\mathcal{A} = (A_1, \ldots, A_n)$, $n \geq 1$, is the distributed reaction system of Γ, $M = (Q, C, R, q_0, F)$ is the context automaton. Let alphabet Σ and mappings ρ, ϕ, ϕ_i, $1 \leq i \leq n$ be defined as in Definition 12.

Let $\pi : \sigma_0 \Longrightarrow \ldots \Longrightarrow \sigma_m$, $m \geq 1$, be a terminating finite interactive process of Γ, where $\sigma_j = (q_j, (c_{j,1}, \ldots, c_{j,n}), (d_{j,1}, \ldots, d_{j,n}))$, $1 \leq j \leq m$, is a state of Γ.

The *concatenation language of* Γ over Σ is defined as follows:

$$L_{concat}(\Gamma, \Sigma, \rho) = \{\phi_1(\sigma_0)\ldots\phi_1(\sigma_m)\ldots\phi_n(\sigma_0)\ldots\phi_n(\sigma_m) \mid \text{ where } \pi : \sigma_0 \Longrightarrow \ldots \Longrightarrow \sigma_m \in \Pi_\Gamma, m \geq 1\}.$$

The *agreement language of* Γ over Σ is defined as follows:

$$L_{agree}(\Gamma, \Sigma, \rho) = \{w \in \Sigma^* \mid w = \phi_i(\sigma_0)\ldots\phi_i(\sigma_m) = \phi_j(\sigma_0)\ldots\phi_j(\sigma_m) \text{ for all } 1 \leq i,j \leq m \text{ and some } \pi : \sigma_0 \Longrightarrow \ldots \Longrightarrow \sigma_m \in \Pi_\Gamma \text{ with } m \geq 1\}.$$

The classes of concatenation and agreement languages of extended distributed reaction systems are denoted by $\mathcal{L}_X(EDRS)$, $X \in \{concat, agree\}$.

In Ciencialová L et al. (2022, 2023) concatenation languages were studied, here we focus on agreement languages.

Theorem 4 $\mathcal{L}_{agree}(\text{ EDRS}) = \bigcup_{k \geq 1} \mathcal{L}(1\text{ NFA }(k))$.

Proof Let $L = L_{agree}(\Gamma, \Sigma, \rho)$ for some extended distributed reaction system $\Gamma = (\Delta, M)$ defined as above, and let us construct the 1NFA(k) $M' = (Q', \Sigma, k, \delta', q'_0, F')$ with

$$Q' = \{q_\sigma \mid q \in Q, \ \sigma \text{ is a state of } \Gamma\} \cup \{q'_0\},$$

$F' = \{q_f\} \not\subseteq F$, and with δ' defined as follows.

Let

$$q_{\sigma_0} \in \delta'(q'_0, \rho(d_1), \ldots, \rho(d_n))$$

for all $\sigma_0 = (q_0, (c_1, \ldots, c_n), (d_1, \ldots, d_n))$ such that q_0 is the initial state of M, and $(c_1, \ldots, c_n) \in C$, $d_i \subseteq S$, $1 \leq i \leq n$.

For any $(c_1, \ldots, c_n) \in C$ and $(d_1, \ldots, d_n)$ with $d_i \subseteq S$, $1 \leq i \leq n$, we also have

$$r_{\sigma_2} \in \delta'(q_{\sigma_1}, \rho(d'_1), \ldots, \rho(d'_n))$$

if and only if

$$\sigma_1 = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n)) \Longrightarrow$$
$$\sigma_2 = (r, (c'_1, \ldots, c'_n), (d'_1, \ldots, d'_n))$$

is a possible transition in Γ. In addition, for all $q \in F$ and σ being a state of Γ, we have

$$\delta'(q_\sigma, \$, \ldots, \$) = \{q_f\}.$$

To see how the 1NFA(k) M' simulates the EDRS Γ, consider the following. An initial state

$$\sigma_0 = (q_0, (c_1, \ldots, c_n), (d_1, \ldots, d_n))$$

of Γ is mapped to the symbols $\phi(\sigma_0) = (\rho(d_1), \ldots, \rho(d_n))$, these will be the first symbols of the languages associated to the components during the computation that will follow. The exact same symbols are read by the multihead automaton M' using one of its initial transitions

$$q_{\sigma_0} \in \delta'(q'_0, \rho(d_1), \ldots, \rho(d_n)).$$

Now, if the initial transition of Γ is

$$\sigma_0 = (q_0, (c_1, \ldots, c_n), (d_1, \ldots, d_n)) \Longrightarrow$$
$$\sigma = (q, (c'_1, \ldots, c'_n), (d'_1, \ldots, d'_n))$$

for some $(q_0, (c_1, \ldots, c_n), q) \in R$ of the context automaton, then the resulting state of this transition is mapped to the symbols $\phi(\sigma) = (\rho(d'_1), \ldots, \rho(d'_n))$, which will also be part of the strings associated to the components of Γ. The same symbols are read by M' using one of its transitions

$$q_\sigma \in \delta'(q_{\sigma_0}, \rho(d'_1), \ldots, \rho(d'_n)).$$

Since the number of possible states $(d_1, \ldots, d_n)$, $d_i \subseteq S$, $1 \le i \le n$, is finite, we have a corresponding transition in M' for any possible state and transition in Γ. If $\sigma = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n))$ is a state (with $\phi(\sigma) = \rho(d_1), \ldots, \rho(d_n)$ being the last symbols of the strings associated to the components so far), then for any transition

$$t: \ \sigma = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n)) \Longrightarrow$$
$$\sigma' = (r, (c'_1, \ldots, c'_n), (d'_1, \ldots, d'_n)),$$

we have

$$r_{\sigma'} \in \delta'(q_\sigma, \rho(d'_1), \ldots, \rho(d'_n))$$

in M' which reads the same symbols as the ones appended to the languages of the components by the transition t above.

Now, if for some state σ, Γ is able to enter a final state (a state like σ' with $r \in F$, as below)

$$\sigma = (q, (c_1, \ldots, c_n), (d_1, \ldots, d_n)) \Longrightarrow$$
$$\sigma' = (r, (c'_1, \ldots, c'_n), (d'_1, \ldots, d'_n)),$$

then M' is able to enter the state $r_{\sigma'}$ by

$$r_{\sigma'} \in \delta'(q_\sigma, \rho(d'_1), \ldots, \rho(d'_n)).$$

If the strings associated to the components of Γ $w_i = \phi_i(\sigma_0)\ldots\phi_i(\sigma_m)$ during the computation $\sigma_0 \Longrightarrow \ldots \Longrightarrow \sigma_m$ coincide, that is, $w = w_i = w_j$ for all $1 \le i,j \le n$, then $w \in L_{agree}(\Gamma, \Sigma, \rho)$, and in this case M' is able to enter into its final state by the transition

$$\delta'(r'_\sigma, \$, \ldots, \$) = \{q_f\}.$$

By the construction of M', we can also see that any state sequence of an accepting computation $q'_0, q_{\sigma_0}, \ldots, q_{\sigma_m}, q_f$, on a string $w \in \Sigma^*$ has a corresponding state sequence

$$\sigma_0 \Longrightarrow \ldots \Longrightarrow \sigma_m$$

in Γ with σ_0 and σ_m being an initial and an accepting state, respectively, and w being the string associated to this computation by the mapping ρ at all of the components.

Let us now assume that L is accepted by a 1NFA(k) $M = (Q, \Sigma, k, \delta, q_0, F)$, and let us construct an extended distributed reaction system $\Gamma = (\Delta, M')$ where $\Delta = (S, \mathcal{A})$ with $\mathcal{A} = (A_1, \ldots, A_k)$ for $A = A_1 = \ldots = A_k$ and

$$S = \Sigma \cup \bar{\Sigma} \cup \{\#, \$\} \text{ where } \bar{\Sigma} = \{\bar{a} \mid a \in \Sigma\},$$
$$A = \{(\{a\}, \{\#\}, \{\bar{a}\}), (\{a, \bar{b}\}, \{\#\}, \{\bar{a}\}),$$
$$(\{\$, \bar{a}\}, \{\#\}, \emptyset) \mid a, b \in \Sigma\}.$$

The context automaton $M' = (Q, C, R, q_0, F)$ is defined as

$$C = \{(\alpha_1, \ldots, \alpha_k) \mid \alpha_i \in \{\emptyset, \{a\} \mid a \in \Sigma\}, 1 \le i \le k\},$$

and for all $q' \in \delta(q, x_1, \ldots, x_k)$, $x_i \in \Sigma \cup \{\lambda\}$, $1 \le i \le k$, we have transitions

$$(q, (s(x_1), \ldots, s(x_k)), q') \text{ where } s(x_i) = \begin{cases} \{x_i\} & \text{if } x_i \neq \lambda, \\ \emptyset & \text{if } x_i = \lambda, \end{cases} \quad 1 \le i \le k,$$

in R. The initial states of the EDRS Γ are those with

$$(q_0, (s(x_1), \ldots, s(x_k)), (\emptyset, \ldots, \emptyset)),$$

such that $\delta(q_0, x_1, \ldots, x_k) \neq \emptyset$, and the mapping $s : \Sigma \cup \bar{\Sigma} \cup \{\lambda\} \to 2^{\Sigma \cup \bar{\Sigma}}$ is defined as above. Thus, the first transitions of Γ are of the form

$$(q_0, (s(x_1), \ldots, s(x_k)), (\emptyset, \ldots, \emptyset))$$
$$\Longrightarrow (q, (s(y_1), \ldots, s(y_k)), (s(\bar{x}_1), \ldots, s(\bar{x}_k)),$$

and they simulate the transitions $q \in \delta(q_0, x_1, \ldots, x_k)$ of M. Now, if a transition $q' \in \delta(q, y_1, \ldots, y_k)$ is also a possible in M, then we have

$$(q, (s(y_1), \ldots, s(y_k)), (s(\bar{x}_1), \ldots, s(\bar{x}_k)) \Longrightarrow$$
$$(q', (\alpha_1, \ldots, \alpha_k), (s(\bar{y}_1), \ldots, s(\bar{y}_k))$$

as the next step of Γ for some $\alpha_i \in 2^\Sigma$, $1 \le i \le k$.

In general, we have

$$(q, (s(x_1), \ldots, s(x_k)), (\beta_1, \ldots, \beta_k) \Longrightarrow (q', (\gamma_1, \ldots, \gamma_k), (s(\bar{x}_1), \ldots, s(\bar{x}_k))$$

for some $\beta_i \in 2^{\bar{\Sigma}}$, $\gamma_i \in 2^{\Sigma}$, $1 \leq i \leq k$, as a possible transition in Γ, if and only if

$$q' \in \delta(q, x_1, \ldots, x_k)$$

$x_i \in \Sigma \cup \{\lambda\}$, is a possible transition in the 1NFA(k) M. Now, if we define the (partial) mapping $\rho : 2^S \to \Sigma \cup \{\lambda\}$ for $\{\bar{a} \mid a \in \Sigma\} \cup \{\emptyset\}$ as $\rho(\{\bar{a}\}) = a$, $a \in \Sigma$, and $\rho(\emptyset) = \lambda$, then we have

$$(x_1, \ldots, x_k) = (\rho(s(\bar{x}_1)), \ldots, \rho(s(\bar{x}_k))),$$

that is, the letters associated to the components of the EDRS Γ as a result of performing the state transition above are the same as the letters read by the heads of k-head automaton M.

Therefore, if

$$(q_0, w\$, \ldots, w\$) \Longrightarrow \ldots \Longrightarrow (q_m, \$, \ldots, \$) \Longrightarrow (q_f, \lambda, \ldots, \lambda)$$

is the sequence of configurations the 1NFA(k) M passes through while accepting an input string $w = a_1 \ldots a_l$, then

$$(q_0, \alpha_1, \ldots, \alpha_k), (\emptyset, \ldots, \emptyset)) \Longrightarrow (q, \beta_1, \ldots, \beta_k), (\bar{\alpha}_1, \ldots, \bar{\alpha}_k)) \Longrightarrow \ldots$$
$$\ldots \Longrightarrow (q_m, \{\$\}, \ldots, \{\$\}), (\bar{\gamma}_1, \ldots, \bar{\gamma}_k)) \Longrightarrow (q_f, \delta_1, \ldots, \delta_k), (\emptyset, \ldots, \emptyset))$$

is a possible computation in Γ with

$$\rho(\emptyset)\rho(\bar{\alpha}_i) \ldots \rho(\bar{\gamma}_i)\rho(\emptyset) = a_1 \ldots a_l = w$$

being the string associated to each component A_i, $1 \leq i \leq k$, of the EDRS Γ.

By the construction of the EDRS Γ we can also see that any terminating interactive process where the same string is associated to each of the components is such, that a corresponding transition sequence accepting the same string exists in the k-head finite automaton M. □

In the following we discuss the languages assigned to qDRS. By definition, it can easily be seen that the behavior of every qDRS Δ is deterministic, i.e., starting from a given initial state $\bar{D}_0$, there exists only one state sequence $\bar{D}_0, \bar{D}_1, \ldots, \bar{D}_m, \ldots$ of Δ. This sequence, as in the case of reaction systems, is either finite or (infinite) ultimately periodic. In Theorem 2 we proved that to any qDRS Δ with an initial state we can construct an EDRS Γ with an initial state such that the state sequence of the qDRS corresponds to the state sequence of the components of the EDRS. Thus, by Definitions 12 and 13, we can assign both agreement and concatenating languages to Γ. If we analyze the proof of Theorem 2, we can easily observe that the language assigned to each component of Γ, thus to each component of Δ is regular. Furthermore, according to the previous remarks, it is either a finite or an ultimately periodic regular language which belongs to a subregular language class. Thus, the agreement language of Δ is either a finite language or an ultimately periodic regular language; both languages are elements of language classes which are proper subclasses of the regular language class and the class of languages accepted by multihead nondeterministic finite automata.

5 Conclusion

In this paper, we introduced a new variant of distributed reaction systems where the component reaction systems communicate with each other by request. This communication protocol differs from the communication protocol of known variants of distributed reaction systems where after performing the reactions, copies of products or reactions are sent to certain predefined target components. That is, the communication is based on command. The behavior of the two types of distributed reaction systems with direct communication, by request or by command, is deterministic. In this paper, we showed that for every distributed reaction system with communication by request a simulating single reaction system can be constructed. We compared the new model to the extended distributed reaction system which works under the control of a context automaton providing input reactants for the components. Extended distributed reaction systems are nondeterministic computing devices. We showed that they simulate the distributed reaction systems with communication by request, but distributed reaction systems with communication by request simulate only a restricted variant of extended distributed reaction systems. We assigned languages to these two variants of distributed reaction systems. We proved that the class of agreement languages of extended distributed reaction systems is equal to the class of languages of multihead nondeterministic finite automata. We discussed the agreement language of the distributed reaction system with communication by request and found that it is an element of a subregular language class.

Author contributions Both authors, EC-V and GV contributed equally to the work.

Funding Open access funding provided by Eötvös Loránd University. No funds, grants, or other support was received.

Data availability No Data associated in the manuscript.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

Ethical approval Not applicable.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Aman B (2022) From networks of reaction systems to communicating reaction systems and back. In: Machines, computations, and universality - 9th international conference, MCU 2022, Debrecen, Hungary, August 31-September 2, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13419, pp. 42–57. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-031-13502-6_3

Aman B (2023) Relating various types of distributed reaction systems. online available in Int J Found Comput Sci. https://www.worldscientific.com/doi/10.1142/S0129054123470044

Azimi S (2017) Steady states of constrained reaction systems. Theor Comput Sci 701:20–26. https://doi.org/10.1016/j.tcs.2017.03.047

Azimi S, Gratie C, Ivanov S, Manzoni L, Petre I, Porreca AE (2016) Complexity of model checking for reaction systems. Theor Comput Sci 623:103–113. https://doi.org/10.1016/j.tcs.2015.11.040

Azimi S, Iancu B, Petre I (2014) Reaction system models for the heat shock response. Fundam Inf 131(3–4):299–312. https://doi.org/10.3233/FI-2014-1016

Bottoni P, Labella A, Rozenberg G (2020) Networks of reaction systems. Int J Found Comput Sci 31(1):53–71

Ciencialová L, Cienciala L, Csuhaj-Varjú E (2022) Languages of distributed reaction systems. In: Machines, Computations, and Universality: 9th International Conference, MCU 2022, Debrecen, Hungary, August 31 - September 2, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13419, pp. 75–90. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-031-13502-6_5

Ciencialová L, Cienciala L, Csuhaj-Varjú E (2023) Language classes of distributed reaction systems. online available in Int J Found Comput Sci https://www.worldscientific.com/doi/pdf/10.1142/S0129054123460024

Csuhaj-Varjú E, Dassow J, Kelemen J, Păun G (1994) Grammar systems: a grammatical approach to distribution and cooperation. Gordon and Breach Science Publisher, Switzerland

Csuhaj-Varjú E, Sethy PK (2020) Communicating reaction systems with direct communication. In: Freund, R., Ishdorj, T., Rozenberg, G., Salomaa, A., Zandron, C. (eds.) Membrane Computing - 21st International Conference, CMC 2020, Virtual Event, September 14-18, 2020, Revised Selected Papers. Lecture Notes in Computer Science, vol. 12687, pp. 17–30. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-030-77102-7_2

Ehrenfeucht A, Main MG, Rozenberg G (2011) Functions defined by reaction systems. Int J Found Comput Sci 22(1):167–178. https://doi.org/10.1142/S0129054111007927

Ehrenfeucht A, Rozenberg G (2007) Reaction systems. Fundam Inf 75(1–4):263–280

Ehrenfeucht A, Rozenberg G (2004) Basic notions of reaction systems. In: Calude C, Calude E, Dinneen MJ (eds) Developments in language theory. In: 8th international conference, DLT 2004, Auckland, New Zealand, December 13–17, 2004, Proceedings, vol 3340. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp 27–29

Hopcroft JE, Motwani R, Ullman JD (2006) Introduction to Automata Theory, Languages, and Computation, 3rd edn. Addison-Wesley Longman Publishing Co., Inc, USA

Meski A, Penczek W, Rozenberg G (2015) Model checking temporal properties of reaction systems. Inf Sci 313:22–42. https://doi.org/10.1016/j.ins.2015.03.048

Meski A, Koutny M, Penczek W (2019) Model checking for temporal-epistemic properties of distributed reaction systems. Tecnical Report Series CS-TR- 1526, Newcastle University

Păun G, Kari L (1989) Parallel communicating grammar systems-The regular case. An Univ Buc Ser Mat-Inform 38(2):55–63

Salomaa A (2012) Functions and sequences generated by reaction systems. Theor Comput Sci 466:87–96. https://doi.org/10.1016/j.tcs.2012.07.022

Salomaa A (2013) Functional constructions between reaction systems and propositional logic. Int J Found Comput Sci 24(1):147–160

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.