Generalized Distributed Reaction Systems

Artiom Alhazov Erzsébet Csuhaj-Varjú
Pramod Kumar Sethy

Abstract

In this paper, we introduce the notion of a generalized distributed reaction system with computations following the concept of the original reaction system: the resulting products in the individual components are obtained by applying rules which take into account the objects in the components of the system as reactants and inhibitors and yield results in specified components of the system. As specific variants, we investigate (i) generalized distributed reaction systems which look at all components for the presence or absence of objects, but the resulting products are only produced in the component the rule is assigned to as well as (ii) generalized distributed reaction systems which look for the presence or absence of objects only in the component the rule is assigned to, but the resulting products can be sent to specified components within the whole system. We first show how all these variants of generalized distributed reaction systems can be flattened to a reaction system having only one component. Moreover, we show how each of these two variants, which are restricted variants of the general model, can simulate even the general model. Finally, we prove that all these variants of generalized distributed reaction systems working with the standard, total parallel application of rules can be transferred into a usual reaction system working with the sequential application of rules.

Keywords: Reaction system, Network of Cells, Flattening
MSC 2020: 68Q01, 68Q07, 68Q10, 68Q85

1 Introduction

Reaction systems (R systems) have extensively been studied over the past two decades, after having been introduced by A. Ehrenfeucht and

doi:10.56415/csjm.v32.17

G. Rozenberg as a formal model for understanding interactions among biochemical reactions. For the original motivation behind this model, we refer to [1]. The primary goal was to model the behavior of biological systems where numerous individual reactions interact with each other. In a reaction system, a reaction is represented as a triplet consisting of reactants (objects required for the reaction to occur), inhibitors (objects whose presence prevents the reaction from taking place), and products (objects produced by the reaction). The dynamic behavior of a reaction system is determined by applying all reactions in parallel which are enabled for the current configuration. Only the union of products obtained in this way constitutes the next state of the system. For more detailed information on reaction systems, see [2].

Reaction systems are a qualitative model, in contrast to P systems (also known as membrane systems, see [3], [4]), which are quantitative. This distinction arises because reaction systems focus solely on the presence or absence of chemical species, without considering their quantities. Reactions that share common reactants do not interfere with one another, and all reactions enabled at a given time step occur simultaneously. Another distinctive feature of reaction systems, compared to other bio-inspired computational models like P systems, is the absence of permanency: the system's current state consists only of the products from the reactions that occurred in the previous time step, while objects not involved in any reaction are eliminated from the system.

Research on reaction systems is quite extensive. Over the past two decades, various properties of reaction systems have been investigated, and numerous extensions have been developed. For instance, studies have addressed fundamental topics such as the concept of time in reaction systems [5], explored dynamic processes and how these processes influence the formation of compounds [6], and examined the mathematical nature of functions (mapping states to states and, therefore, finite sets to finite sets) that can be defined by reaction systems [7]. In [8]–[10], several mathematical aspects of reaction systems were explored, including functions defined by these systems, the properties of their state sequences, the impact of limited resources, and their connections to propositional logic. Another research direction focuses on the

use of reaction systems as a modeling framework. For verifying temporal properties in reaction systems, a temporal logic was introduced in [11]. Biologically inspired characteristics of reaction systems were studied in [12]–[14]. Additionally, reaction systems can be structured within a distributed communication framework [15], [16]. In this context, reactions are positioned at the nodes of virtual graphs, or more specifically, represented as an n-tuple of reaction systems. These reactions operate synchronously and interact through distribution and communication protocols. Interaction can take place either by receiving input from an external environment or through communication between reactants, inhibitors, products, or reactions within the systems. Examples of such frameworks include distributed reaction systems [11], extended distributed reaction systems [17], [18], communicating reaction systems with direct communication [19], and further variants of distributed reaction systems [20], [21]. Further developments of these constructs can be found in [22], [23].

In this paper, we introduce the concept of the generalized distributed reaction system with computations following the concept of the original reaction system, i.e., the resulting products in the individual cells are obtained by applying reaction rules that take into account the objects in all the components of the system as reactants and inhibitors; the obtained results may be sent to any component.

As specific variants of this general model of generalized distributed reaction systems, we investigate

(i) generalized distributed reaction systems which look at all components for the presence or absence of objects, but the resulting products are only produced in the component the rule is assigned to as well as

(ii) generalized distributed reaction systems which look for the presence or absence of objects only in the cell the rule is assigned to, but the resulting products can be sent to specified components within the whole system.

As a basic result, we will show how all these variants of generalized distributed reaction systems can be flattened to a reaction system,

which corresponds to a generalized distributed reaction system having only one component. Moreover, we elaborate on the simulation of each of these two variants by the other one. Finally, we shall prove that all these variants of reaction systems working with the total parallel application of rules according to the basic model of reaction systems can be transferred into a usual reaction system working with the sequential application of rules, i.e., applying only one rule in every computation step.

The paper is structured as follows: Section 2 reviews key concepts from formal language theory and reaction systems. In Section 3, we first introduce the model of generalized distributed reaction systems and its two variants along with some small examples. Then we elaborate our results concerning flattening and the simulation of one variant by the other one as well as the results for generalized distributed reaction systems working in the sequential derivation mode. Finally, we provide a conclusion of the results obtained in this paper and propose some directions for future research.

2 Preliminaries

For basic notions and notations in formal language and computation theory, the reader is referred to textbooks like [24].

The cardinality of a set M is denoted by $|M|$, the empty set – by $\emptyset$. The set of natural numbers is denoted by $\mathbb{N}$, the set of positive natural numbers – by $\mathbb{N}_+$.

We now recall the key concepts related to reaction systems from [1], [2]. Note that some notations may have slightly been modified from the ones appearing in other papers for technical reasons.

Definition 1. *Let S be a finite, nonempty set, referred to as the background set. A* reaction *over S is defined as a triplet $\rho = (R, I, P)$, where R, I, and P are nonempty subsets of S with the condition that $R \cap I = \emptyset$. The sets R, I, and P are referred to as the set of* reactants, *the set of* inhibitors, *and the set of* products *of ρ, respectively, and can also be denoted by R_ρ, I_ρ, and P_ρ.*

Definition 2. *A reaction system is defined as an ordered pair* $\Pi = (S, A)$*, where* S *is the background set, and* A *is a finite, non-empty set of reactions over* S*.*

Next, we define the effect of a reaction given a specific *state* or *configuration* of a reaction system, i.e., on a finite subset of S.

Definition 3. *Let* $\Pi = (S, A)$ *be a reaction system with* S *being the background set and* A *a set of reactions over* S *as well as* $T \subseteq S$ *and* $\rho = (R_\rho, I_\rho, P_\rho) \in A$ *a reaction over* S*. Then*

1. ρ *is enabled for* T *if and only if* $R_\rho \subseteq T$ *and* $I_\rho \cap T = \emptyset$*.*
2. *The result of applying* ρ *to* T*, denoted* $res_\rho(T)$*, is* P_ρ *if* ρ *is enabled for* T*, and* $\emptyset$ *otherwise.*
3. *The result of applying* A *to* T*, denoted* $res_A(T)$*, is* $\bigcup_{\rho \in A} res_\rho(T)$*;* res_A *defines a function on* 2^S*, called the result function.*

Thus, reaction ρ is enabled for T if T includes all reactants of ρ and none of its inhibitors. When ρ is enabled for T, it contributes its products to the successor state. For $T \subseteq S$, $en_A(T)$ denotes the set of reactions from A that are enabled for T. If no confusion arises, we may use $en_\rho(T)$ instead of $en_{\{\rho\}}(T)$, where ρ is a single reaction.

The state sequence of a reaction system ρ with initial state T is given by successive iterations of the result function, with $res_A^0(T) = T$:

$$(res_A^n(T))_{n \in \mathbb{N}} = (T, res_A(T), res_A^2(T), \dots).$$

Remark 1. *In the usual definition of reaction systems, the set* R_ρ *in a reaction* $\rho = (R_\rho, I_\rho, P_\rho) \in A$ *is required to be non-empty, which means that a state sequence yielding the empty set ends without successor state. Yet, to be consistent with the generalized model introduced later, we avoid pecularities with disallowing this special case. Moreover, we also allow the result set* P_ρ *to be the empty set, which anyway does not contribute to the result set of the whole system.*

It is important to note that, because the background set of a reaction system is finite, the state space, i.e., the number of possible

configurations, is also finite. As a result, every state sequence either is finite, ending up with the empty set, or eventually becomes periodic. Therefore, usually the behavior of reaction systems is considered as an interactive process, where the reaction system gets input from the environment.

Definition 4. *Let $\Pi = (S, A)$ be a reaction system with S being the background set and A a set of reactions over S. An interactive process with Π considers an initial set T and a sequence of inputs*

$$(D_0, D_1, \dots), \quad D_i \subseteq S, \ i \geq 0,$$

thus yielding the sequence of results

$$(R_0, R_1, \dots), \quad R_i, \ i \geq 0,$$

with $R_0 = T$ and $R_i = res_A(D_{i-1} \cup R_{i-1})$, $i \geq 1$.

Example 1. *Consider the reaction system $\Pi = (S, A)$, $S = \{a, b\}$, $A = \{\rho_1\} = (\{a\}, \{b\}, \{b\})\}$.*

With $T = \{a\}$, but without input sequence, we would only obtain the finite state sequence $(\{a\}, \{b\}, \emptyset)$, as ρ_1 is not enabled for $\{b\}$. Yet, together with the input sequence

$$(D_0, D_1, \dots), \quad D_i = \{a\}, \ i \geq 0,$$

we obtain the result sequence

$$(R_0, R_1, \dots), \quad R_i, \ i \geq 0,$$

with $R_{2n} = \{a\}$, $R_{2n+1} = \{a, b\}$ and $R_{2n+2} = \emptyset \cup \{a\} = R_{2n}$, for all $n \geq 0$, as ρ_1 applied to R_{2n} yields $\{b\}$, ρ_1 cannot be applied to $R_{2n+1} = \{a, b\}$, yielding the empty set as result.

3 Generalized Distributed Reaction Systems

Reaction systems can be generalized to work as a distributed and communicating system. As we mentioned in the Introduction, in these models reactions are positioned at the nodes of virtual graphs, or more

specifically, represented as an n-tuple of reaction systems. These reactions operate synchronously and interact through distribution and communication protocols. For example, communicating reaction systems with direct communication (cdcR systems for short) are given by an $(n+1)$-tuple $\Pi = (S, A_1, \ldots, A_n)$, where S is a background set and $A_1, \ldots, A_n$ are sets of reactions over S, the components of Π [19]. Two variants of cdcR systems have been examined: cdcR systems, where the components, after performing the enabled reactions, communicate copies of the obtained products to designated components, and cdcR systems, where the components, after performing the enabled reaction, communicate copies of the reaction itself to some designated components. In [19], it was shown that both types of cdcR systems can be flattened, i.e., to be converted to a single reaction system. That is, the state sequence of the cdcR system and that of the R system correspond to each other.

In part, these results and proof techniques have inspired the development of the notion of the generalized distributed reaction system, which provides a common framework for studying variants of distributed reaction systems.

We first define the general model and then consider the two restricted variants already mentioned in the Introduction.

Definition 5. *A* generalized distributed reaction system *(a* GDR system *for short) is a triple* (n, S, A)*, where* n *is the number of components labeled* $1, \cdots, n$*,* S *is the background set,* $E = \{1, \cdots, n\} \times S$ *is defined as the extended alphabet, i.e., pairs of component index and object, and* A *is the set of reactions of the form* $\rho = (R_\rho, I_\rho, P_\rho)$*, where* R_ρ, I_ρ, P_ρ *are subsets of* E*, called* reactants*,* inhibitors*, and* products*, respectively. We write elements of* E *as* $[i, a]$*, where* $1 \leq i \leq n$ *and* $a \in S$*.*

In the following, we use notation $\text{Component}([i, a]) = i$ and $\text{Component}(T) = \{\text{Component}(t) \mid t \in T\}$.

A configuration C is a subset of E, which can be split into n components, resulting in an n-tuple $(C_1, \ldots, C_n)$ of subsets of S with $C_i = \{[i, a] \mid [i, a] \in C, a \in S\}$. We then may also write $C_i = \{a \mid [i, a] \in C, a \in S\}$.

A reaction $\rho = (R_\rho, I_\rho, P_\rho)$ is enabled if and only if each reactant

in R_ρ is present and each inhibitor in I_ρ is absent, yielding P_ρ as a result. The next configuration is the set union of products of all enabled reactions.

3.1 Restricted Variants of GDR Systems

We now consider the two special variants of GDR systems already mentioned above:

(i) generalized distributed reaction systems which look at all components for the presence or absence of objects, but the resulting products are only produced in the component the rule is assigned to as well as

(ii) generalized distributed reaction systems which look for the presence or absence of objects only in the component the rule is assigned to, but the resulting products can be sent to specified components within the whole system.

In case (i), we denote such a GDR system as a *GDR system with total checks and local output* (a *$GDR_{t,l}$ system* for short), in case (ii), as a *GDR system with local checks and total output* (a *$GDR_{l,t}$ system* for short).

In both cases, the GDR system (n, S, A) can be represented as $\Delta = (n, S, \mathcal{A})$ with $\mathcal{A}$ consisting of n components $\mathcal{A}_i$, $1 \leq 1 \leq n$, i.e., $\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_n)$.

(i) For a $\mathrm{GDR}_{t,l}$ system, we get

$$\mathcal{A}_i = \{[i, a] \mid [i, a] \in P_\rho \text{ for some reaction } \rho \in \mathrm{A}\};$$

observe that, for every reaction $\rho \in A$, all results must go to the same component i, i.e., $|\mathrm{Component}(P_\rho)| = 1$ and $\mathrm{Component}(P_\rho) = i$; and

(ii) for a $\mathrm{GDR}_{t,l}$ system, we get

$$\mathcal{A}_i = \{[i, a] \mid [i, a] \in R_\rho \cap I_\rho \text{ for some reaction } \rho \in \mathrm{A}\},$$

i.e., we collect all reaction rules with reactants and inhibitors only in component i; observe that for every reaction $\rho \in A$ all reactants and inhibitors must be from the same component, i.e., $|\mathrm{Component}(R_\rho \cup I_\rho)| = 1$ and $\mathrm{Component}(R_\rho \cup I_\rho) = i$.

We mention that these two variants have been considered as variants of communicating reaction systems. $\mathrm{GDR}_{l,t}$ systems were called communicating reaction systems with direct communication (cdcR systems) where products are communicated (cdcR(p) systems) [19]. Predecessor models of $\mathrm{GDR}_{t,l}$ systems, the cdcR(c) systems, represent indirect communication. Namely, the execution of a reaction also requires checking the presence/absence of objects in some designated nodes of the communicating reaction system [25], [26].

We now give two short examples for these special variants of GDR systems, yielding simple periodic sequences of configurations:

Example 2. *Consider the* $GDR_{t,l}$ *system* $\Delta = (2, \{a, b\}, \mathcal{A})$ *with* $\mathcal{A}_1 = \{(\{[1, a], [2, b]\}, \emptyset, \{[1, a]\})\}$ *and* $\mathcal{A}_2 = \{(\{[1, a], [2, b]\}, \emptyset, \{[2, b]\})\}$. *Starting with the initial configuration* $C_0 = (\{[1, a]\}, \{[2, b]\}$, *by applying reaction rule* $(\{[1, a], [2, b]\}, \emptyset, \{[1, a]\})$ *in the first component and reaction rule* $(\{[1, a], [2, b]\}, \emptyset, \{[2, b]\})$ *in the second component, we again obtain the same configuration, i.e., the periodic sequences of configurations* $(C_0, C_1, \dots)$ *with* $C_i = (\{[1, a]\}, \{[2, b]\})$ *for* $i \geq 0$*; we observe that these configurations can also be represented in a simpler way by* $C_i = (\{a\}, \{b\})$.

Example 3. *Consider the* $GDR_{l,t}$ *system* $\Delta = (2, \{a, b\}, \mathcal{A})$ *with* $\mathcal{A}_1 = \{(\{[1, a]\}, \emptyset, \{[1, a], [2, b]\})\}$ *and* $\mathcal{A}_2 = \emptyset$. *Starting with the initial configuration* $C_0 = (\{[1, a]\}, \emptyset)$, *i.e.,* $C_0 = (\{a\}, \emptyset)$, *by applying reaction rule* $(\{[1, a]\}, \emptyset, \{[1, a], [2, b]\})$ *in the first component, we obtain the configuration* $C_1 = (\{[1, a]\}, \{[2, b]\}) = (\{a\}, \{b\})$*; in the next computation steps, we obtain the same configuration again, i.e., from the initial configuration, we get the periodic sequences of configurations* $(C_1, C_2, \dots)$ *with* $C_i = (\{[1, a]\}, \{[2, b]\}) = (\{a\}, \{b\})$ *for* $i \geq 1$.

3.2 Flattening of GDR Systems

The following theorem shows that GDR system (n, S, A) can be *flattened* to a corresponding standard reaction system (E, A).

Theorem 1. *Every GDR system* $(n, S, \mathcal{A})$ *can be converted into the corresponding standard reaction system* (E, A).

Proof: We set $E = \{1, \cdots, n\} \times S$. The extended alphabet simply becomes the background set in the new system. The computation in the reaction system (E, A) is isomorphic to the computation in the original GDR system $(n, S, \mathcal{A})$, with the object a in component i precisely corresponding to object $[i, a]$ in the reaction system (E, A). □

Hence, the generalization of reaction systems to generalized distributed reaction systems does not increase the power of reaction systems. Moreover, we immediately infer that the same flattening procedure also works for the the special cases of $\mathrm{GDR}_{t,l}$ and $\mathrm{GDR}_{l,t}$ systems. Therefore, we also infer that these variants can simulate each other. However, to recover original sequences of configurations, one needs to unpack the objects a from the extended objects $[i, a]$ and group them in the respective regions i.

As for standard reaction systems, GDR systems also can be considered with input streams

- of symbols from the extended alphabet and they can enter the corresponding components;
- of symbols from the background alphabet S which can only enter a designated input component.

The computations of GDR systems with input streams can be defined as for standard reaction systems, especially when looking at the corresponding flattened systems, see Theorem 1. Yet, in this paper, we refrain from giving a formal definition and continue our investigations without considering the dynamics coming along with input sequences.

3.3 Simulation of GDR Systems by Restricted Variants of GDR Systems

We now show how any GDR system $(n, S, \mathcal{A})$ can be simulated by a corresponding $\mathrm{GDR}_{l,t}$ system as well as by a corresponding $\mathrm{GDR}_{t,l}$ system in such a way that in the first n components in these restricted systems any computation coincides with the first n components of the original GDR system. The main idea of the constructions elaborated in the proof of the following theorem is to simulate the flattened system of the given GDR system $(n, S, \mathcal{A})$ according to Theorem 1 in an additional component.

Theorem 2. *Every GDR system $(n, S, \mathcal{A})$ can be simulated by a corresponding $GDR_{l,t}$ system as well as by a corresponding $GDR_{t,l}$ system, respectively, with the first n components in any computation coinciding with the first components of the original GDR system.*

Proof: We start with an arbitrary GDR system $(n, S, \mathcal{A})$, which we can flatten according to Theorem 1, thus obtaining a reaction system (E, A).

We now construct the corresponding $\mathrm{GDR}_{l,t}$ and $\mathrm{GDR}_{t,l}$ system, respectively, with $n+1$ components, where, in component $n+1$, the flattened reaction system (E, A) is simulated. Note that each object $[i, a]$ from the set of products of a reaction in A, when placed in component $n+1$, is to be written as $[[i, a], n+1]$. Now we have two cases, depending on what kind of system we construct.

Constructing the $\mathbf{GDR}_{l,t}$ system

Of course, in the initial configuration, we have the first n components to match those in the original system. For the subsequent steps, component $n+1$ already performs the complete computation for the whole system, so it suffices to also unpack a copy of each object, and send it to the correct component. Hence, we replace each reaction

$$(R, I, \{[[i_1, a_1], n+1], \cdots, [[i_k, a_k], n+1]\}) \in \mathcal{A}_{n+1}$$

by the corresponding reaction in component $n+1$

$$(R, I, \{[[i_1, a_1], n+1], \cdots, [[i_k, a_k], n+1], [i_1, a_1], \cdots, [i_k, a_k]\}).$$

To the first n components no reaction rules need to be assigned, everything only happens in the additional component $n+1$.

Constructing the $\mathbf{GDR}_{l,t}$ system

Also in this case, we want the first n components of the initial configuration to match those in the original system. For the subsequent steps, component $n+1$ will already perform the complete computation, so it suffices to import (a copy of) each object by the corresponding

component. Hence, now every component i, $1 \leq i \leq n$, has to have reactions $(\{[[i, a], n+1]\}, \emptyset, \{a\})$ for each object $a \in S$.

Yet, in order to make this simulation realtime, component $n+1$ must contain the results of a one-step look-ahead computation, i.e., in the mth configuration C_m, $m \geq 0$, of a computation sequence $C_0, C_1, \ldots$, component $n+1$ must already contain the result of computation step $m+1$ to allow the other components to get the correct objects in this computation step, which, for the initial configuration, means that component $n+1$ already contain the result of the first computation in the flattened system. □

Therefore, $\mathrm{GDR}_{t,l}$ systems and $\mathrm{GDR}_{l,t}$ systems can simulate arbitrary GDR systems and, hence, each other, retaining the original values in the first n components of any sequence of configurations, both with using one additional component.

3.4 Sequential GDR Systems

Let us call the derivation mode when all enabled reactions are applied in parallel *total parallelism*. We claim that this total parallelism used in reaction systems can be simulated by a *sequential* reaction system, i.e., when at most one reaction is enabled in any step.

Theorem 3. *Every GDR system $(n, S, \mathcal{A})$ can be converted into the corresponding sequential reaction system (E, A).*

Proof: We start with applying Theorem 1 to obtain the corresponding flattened reaction system (E, A') from the given GDR system $(n, S, \mathcal{A})$. Hence, we only have to construct the corresponding sequential reaction system $\Pi = (E, A)$ for $\Pi' = (E, A')$. Now consider the function $f : 2^E \to 2^E$ defined as $f(T) = T'$ if T' can be derived from T in Π, for any $T \subseteq E$. We claim that function f can be implemented by the sequential reaction system $\Pi = (E, A)$, as we add to A reaction $(T, E \setminus T, T')$. Note that this reaction is only enabled for configuration T, and it produces the desired result. □

The proof of the preceding theorem immediately infers the following even stronger result for standard reaction systems.

Corollary 1. *Every reaction system* (S, A) *can be converted into the corresponding sequential reaction system* (E, A) *with even preserving the sequences of configurations.*

4 Conclusions

In this article, we extended the concept of reaction systems to generalized distributed reaction systems (GDR systems), where the resulting products in the individual components are obtained by applying rules which take into account the objects in all the components of the system as reactants and inhibitors and yield results in specified components of the system. As specific variants, we have investigated for the presence or absence of objects, but the resulting products are only produced in the component the rule is assigned to, as well as (ii) GDR systems which look for the presence or absence of objects only in the component the rule is assigned to, but the resulting products can be sent to specified components within the whole system.

All these variants of GDR systems can be flattened to a standard reaction system. Moreover, we showed how each of these two variants can simulate each other with corresponding state sequences. Furthermore, GDR systems working with the total parallel application of rules can be transferred into a usual reaction system working with the sequential application of rules.

In [27], [28], occurrence problems and their complexity for reaction systems have been studied and were shown to be NP-complete (or PSPACE-complete) problems, depending on how the problem is formulated. Such an analysis should also be done for GDR systems and its variants as, for example, already discussed partly for the communication dynamics within the system for $\mathrm{GDR}_{l,t}$ systems in [19].

Although the variants of GDR systems discussed in this paper show that GDR systems do not go beyond the boundaries of reaction systems, there are interesting problems that need to be studied. One such research topic could be: to investigate which communication protocols might be reasonable to define and to study the different semantics associated with them. It would also be of interest to investigate how the input streams of the GDR system can be classified, and which type of

input streams has a significant impact on the configuration sequence of the GDR system, and how its significance can be measured.

Acknowledgments. The authors are grateful to Rudolf Freund for his inspiring ideas in the development of the idea of generalizing the concept of reaction systems and for his many valuable comments and suggestions on the drafts during the preparation of the paper. His contribution is invaluable for this paper and the future work in this topic. Artiom Alhazov also acknowledges Project 011301 "Information systems based on artificial intelligence" by the Moldova State University.

References

[1] A. Ehrenfeucht and G. Rozenberg, "Basic notions of reaction systems," in *Developments in Language Theory, 8th International Conference, DLT 2004, Auckland, New Zealand, December 13-17, 2004, Proceedings*, ser. Lecture Notes in Computer Science, C. Calude, E. Calude, and M. J. Dinneen, Eds., vol. 3340. Springer, 2004, pp. 27–29.

[2] A. Ehrenfeucht and G. Rozenberg, "Reaction systems," *Fundam. Informaticae*, vol. 75, no. 1-4, pp. 263–280, 2007.

[3] Gh. Păun, "Computing with membranes," *Journal of Computer and System Sciences*, vol. 61, no. 1, pp. 108–143, 2000.

[4] Gh. Păun, G. Rozenberg, and A. Saloma, *The Oxford Handbook of Membrane Computing.* Oxford University Press,Oxford, 2010.

[5] A. Ehrenfeucht and G. Rozenberg, "Introducing time in reaction systems," *Theor. Comput. Sci.*, vol. 410, no. 4-5, pp. 310–322, 2009.

[6] A. Ehrenfeucht and G. Rozenberg, "Events and modules in reaction systems," *Theor. Comput. Sci.*, vol. 376, no. 1-2, pp. 3–16, 2007.

[7] A. Ehrenfeucht, M. Main, and G. Rozenberg, "Combinatorics of life and death for reaction systems," *International Journal of*

Foundations of Computer Science, vol. 21, no. 03, pp. 345–356, 2010.

[8] A. Ehrenfeucht, M. G. Main, and G. Rozenberg, "Functions defined by reaction systems," *Int. J. Found. Comput. Sci.*, vol. 22, no. 1, pp. 167–178, 2011.

[9] A. Salomaa, "Functions and sequences generated by reaction systems," *Theor. Comput. Sci.*, vol. 466, pp. 87–96, 2012.

[10] A. Salomaa, "Functional constructions between reaction systems and propositional logic," *Int. J. Found. Comput. Sci.*, vol. 24, no. 1, pp. 147–160, 2013.

[11] A. Meski, M. Koutny, and W. Penczek, "Model checking for temporal-epistemic properties of distributed reaction systems," Newcastle University, Tecnical Report Series CS-TR- 1526, April 2019.

[12] S. Azimi, "Steady states of constrained reaction systems," *Theor. Comput. Sci.*, vol. 701, pp. 20–26, 2017.

[13] S. Azimi, C. Gratie, S. Ivanov, and I. Petre, "Dependency graphs and mass conservation in reaction systems," *Theor. Comput. Sci.*, vol. 598, pp. 23–39, 2015.

[14] S. Azimi, B. Iancu, and I. Petre, "Reaction system models for the heat shock response," *Fundam. Informaticae*, vol. 131, no. 3-4, pp. 299–312, 2014.

[15] P. Bottoni, A. Labella, and G. Rozenberg, "Reaction systems with influence on environment," *J. Membr. Comput.*, vol. 1, no. 1, pp. 3–19, 2019.

[16] P. Bottoni, A. Labella, and G. Rozenberg, "Networks of reaction systems," *Int. J. Found. Comput. Sci.*, vol. 31, no. 1, pp. 53–71, 2020.

[17] L. Ciencialová, L. Cienciala, and E. Csuhaj-Varjú, "Languages of distributed reaction systems," in *Machines, Computations, and Universality - 9th International Conference, MCU 2022, Debrecen,*

Hungary, August 31 - September 2, 2022, Proceedings, ser. Lecture Notes in Computer Science, J. Durand-Lose and Gy. Vaszil, Eds., vol. 13419. Springer, 2022, pp. 75–90.

[18] L. Ciencialová, L. Cienciala, and E. Csuhaj-Varjú, "Language classes of extended distributed reaction systems," *International Journal of Foundations of Computer Science*, pp. 1–24, 2023. [Online]. Available: https://www.worldscientific.com/doi/pdf/10.1142/S0129054123460024

[19] E. Csuhaj-Varjú and P. K. Sethy, "Communicating reaction systems with direct communication," in *Membrane Computing – 21st International Conference CMC 2020, Vienna, Austria, September 14-18, 2020, Revised Selected Papers*, ser. Lecture Notes in Computer Science, R. Freund, T.-O. Ishdorj, G. Rozenberg, A. Salomaa, and C. Zandron, Eds., vol. 12687. Springer, 2021, pp. 17–30.

[20] A. Bagossy and Gy. Vaszil, "Controlled reversibility in communicating reaction systems," *Theor. Comput. Sci.*, vol. 926, pp. 3–20, 2022.

[21] E. Csuhaj-Varjú and Gy. Vaszil, "Variants of distributed reaction systems," *Nat. Comput.*, vol. 23, no. 2, pp. 269–284, 2024.

[22] B. Aman, "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*, ser. Lecture Notes in Computer Science, J. Durand-Lose and Gy. Vaszil, Eds., vol. 13419. Springer, 2022, pp. 42–57.

[23] B. Aman, "Relating various types of distributed reaction systems," *International Journal of Foundations of Computer Science*, pp. 1–16, 2023. [Online]. Available: https://www.worldscientific.com/doi/10.1142/S0129054123470044

[24] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to automata theory, languages, and computation, 3rd Edition*, ser. Pearson international edition. Addison-Wesley, 2007.

[25] E. Csuhaj-Varjú and P. K. Sethy, "Direct and indirect communication in communicating reaction systems," 2024, manuscript.

[26] P. K. Sethy, "Communication in networks of bio-inspired computing systems," Submitted PhD dissertation, Doctoral School of Informatics, Eötvös Loránd University, Budapest, 2024.

[27] E. Formenti, L. Manzoni, and A. E. Porreca, "On the complexity of occurrence and convergence problems in reaction systems," *Nat. Comput.*, vol. 14, no. 1, pp. 185–191, 2015.

[28] A. Salomaa, "Functional constructions between reaction systems and propositional logic," *Int. J. Found. Comput. Sci.*, vol. 24, no. 1, pp. 147–160, 2013.

Artiom Alhazov,
Erzsébet Csuhaj-Varjú,
Pramod Kumar Sethy

Received October 14, 2024
Revised October 18, 2024
Accepted October 23, 2024

Artiom Alhazov
ORCID: https://orcid.org/0000-0002-6184-3971
State University of Moldova,
Vladimir Andrunachievici Institute of Mathematics and Computer Science
Academiei 5, Chișinău, MD-2028, Moldova
E–mail: artiom@math.md

Erzsébet Csuhaj-Varjú
ORCID: https://orcid.org/0000-0002-2773-2944
Department of Algorithms and Their Applications, Faculty of Informatics, Eötvös Loránd University
Pázmány Péter sétány 1/c, Budapest, 1117, Hungary
E–mail: csuhaj@inf.elte.hu

Pramod Kumar Sethy
ORCID: https://orcid.org/0000-0002-3617-0920
Department of Algorithms and Their Applications, Faculty of Informatics, Eötvös Loránd University
Pázmány Péter sétány 1/c, Budapest, 1117, Hungary
E–mail: pksethy@inf.elte.hu