数理解析研究所講究録
第 1799 巻 2012 年 179-182

Automata inspired by biochemical reaction*

早稲田大学大学院 教育学研究科　大久保 文哉 (Fumiya Okubo)
Graduate School of Education, Waseda University
電気通信大学大学院 電気通信学研究科　小林 聡 (Satoshi Kobayashi)
Graduate School of Informatics and Engineering, University of Electro-Communications
早稲田大学 教育・総合科学学術院　横森 貴 (Takashi Yokomori)
Department of Mathematics, Faculty of Education and Integrated Arts and Sciences,
Waseda University

1 Introduction

Motivated by two notions of a reaction system ([3, 4, 5]) and a multiset ([1]), in this paper we will introduce computing devices called *reaction automata* and show that they are computationally universal by proving that any recursively enumerable language is accepted by a reaction automaton. There are two points to be remarked: On one hand, the notion of reaction automata may be taken as a kind of an extension of reaction systems in the sense that our reaction automata deal with *multisets* rather than (usual) sets as reaction systems do, in the sequence of computational process. On the other hand, however, reaction automata are introduced as computing devices that accept the sets of *string objects* (i.e., languages over an alphabet). This unique feature, i.e., a string accepting device based on multiset computing in the biochemical reaction model can be realized by introducing a simple idea of feeding an input to the device from the environment.

This paper is organized as follows. After preparing the basic notions and notations in Section 2, we introduce the main notion of reaction automata together with one language example in Section 3. Moreover we present our main results: reaction automata are computationally universal. We also consider some subclasses of reaction automata from a viewpoint of the complexity theory in Section 4, and investigate the language classes accepted by those subclasses in comparison to the Chomsky hierarchy. Finally, concluding remarks as well as future research topics are discussed in Section 5.

2 Preliminaries

We assume that the reader is familiar with the basic notions of formal language theory. For unexplained details, refer to [8].

We use the basic notations regarding multisets that follow [2, 9]. A *multiset* over an alphabet V is a mapping $\mu : V \to \mathbf{N}$, where $\mathbf{N}$ is the set of non-negative integers and for each $a \in V$, $\mu(a)$ represents the number of occurrences of a in the multiset μ. The set of all multisets over V is denoted by $V^{\#}$, including the empty multiset denoted by μ_λ, where $\mu_\lambda(a) = 0$ for all $a \in V$. We often identify a multiset μ with its string representation $w_\mu = a_1^{\mu(a_1)} \cdots a_n^{\mu(a_n)}$ or any permutation of w_μ.

*The work of F. Okubo was in part supported by Waseda University Grant for Special Research Projects: 2011A-842. The work of S. Kobayashi was in part supported by Grants-in-Aid for Scientific Research (C) No.22500010, Japan Society for the Promotion of Science. The work of T. Yokomori was in part supported by Waseda University Grant for Special Research Projects: 2011B-056.

A usual set $U \subseteq V$ is regarded as a multiset μ_U such that $\mu_U(a) = 1$ if a is in U and $\mu_U(a) = 0$ otherwise. In particular, for each symbol $a \in V$, a multiset $\mu_{\{a\}}$ is often denoted by a itself.

For two multisets μ_1, μ_2 over V, we define one relation and three operations as follows:

$$\begin{array}{ll} \textit{Inclusion}: & \mu_1 \subseteq \mu_2 \text{ iff } \mu_1(a) \leq \mu_2(a), \\ \textit{Sum}: & (\mu_1 + \mu_2)(a) = \mu_1(a) + \mu_2(a), \\ \textit{Intersection}: & (\mu_1 \cap \mu_2)(a) = \min\{\mu_1(a), \mu_2(a)\}, \\ \textit{Difference}: & (\mu_1 - \mu_2)(a) = \mu_1(a) - \mu_2(a), \\ & \text{(for the case } \mu_2 \subseteq \mu_1 \text{ only)}, \end{array}$$

for each $a \in V$. The sum for a family of multisets $\mathcal{M} = \{\mu_i\}_{i \in I}$ is denoted by $\sum_{i \in I} \mu_i$. For a multiset μ and $n \in \mathbf{N}$, μ^n is defined by $\mu^n(a) = n \cdot \mu(a)$ for each $a \in V$. The *weight* of a multiset μ is $|\mu| = \sum_{a \in V} \mu(a)$.

3 Reaction Automata

By recalling from [3] basic notions related to reactions systems, we first extend them (defined on the sets) to the notions on the multisets. Then, we shall introduce our notion of *reaction automata* which plays a central role in this paper.

Definition 1. For a set S, a *reaction* in S is a 3-tuple $\mathbf{a} = (R_\mathbf{a}, I_\mathbf{a}, P_\mathbf{a})$ of finite multisets, such that $R_\mathbf{a}, P_\mathbf{a} \in S^\#$, $I_\mathbf{a} \subseteq S$ and $R_\mathbf{a} \cap I_\mathbf{a} = \emptyset$.

The multisets $R_\mathbf{a}$ and $P_\mathbf{a}$ are called the *reactant* of $\mathbf{a}$ and the *product* of $\mathbf{a}$, respectively, while the set $I_\mathbf{a}$ is called the *inhibitor* of $\mathbf{a}$. These notations are extended to a multiset of reactions as follows: For a set of reactions A and a multiset α over A,

$$R_\alpha = \sum_{\mathbf{a} \in A} R_\mathbf{a}^{\alpha(\mathbf{a})},\ I_\alpha = \bigcup_{\mathbf{a} \subseteq \alpha} I_\mathbf{a},\ P_\alpha = \sum_{\mathbf{a} \in A} P_\mathbf{a}^{\alpha(\mathbf{a})}.$$

Definition 2. Let A be a set of reactions in S and $\alpha \in A^\#$ be a multiset of reactions over A. Then, for a finite multiset $T \in S^\#$, we say that
(1) α is *enabled by* T if $R_\alpha \subseteq T$ and $I_\alpha \cap T = \emptyset$,
(2) α is *enabled by* T *in maximally parallel manner* if there is no $\beta \in A^\#$ such that $\alpha \subset \beta$, and α and β are enabled by T.
(3) By $En_A^p(T)$ we denote the set of all multisets of reactions $\alpha \in A^\#$ which are enabled by T in maximally parallel manner.
(4) The *results of* A *on* T, denoted by $Res_A(T)$, is defined as follows:

$$Res_A(T) = \{T - R_\alpha + P_\alpha \mid \alpha \in En_A^p(T)\}.$$

Note that we have $Res_A(T) = \{T\}$ if $En_A^p(T) = \emptyset$.

Definition 3. (Reaction Automata) A *reaction automaton* (RA) $\mathcal{A}$ is a 5-tuple $\mathcal{A} = (S, \Sigma, A, D_0, f)$, where

- S is a finite set, called the *background set of* $\mathcal{A}$,
- $\Sigma (\subseteq S)$ is called the *input alphabet of* $\mathcal{A}$,
- A is a finite set of reactions in S,
- $D_0 \in S^\#$ is an *initial multiset*,
- $f \in S$ is a special symbol which indicates the final state.

Definition 4. Let $\mathcal{A} = (S, \Sigma, A, D_0, f)$ be an RA and $w = a_1 \cdots a_n \in \Sigma^*$. An *interactive process in* $\mathcal{A}$ *with input* w is an infinite sequence $\pi = D_0, \ldots, D_i, \ldots$, where

$$\begin{cases} D_{i+1} \in Res_A(a_{i+1} + D_i) & (\text{for } 0 \leq i \leq n-1), \\ D_{i+1} \in Res_A(D_i) & (\text{for all } i \geq n). \end{cases}$$

By $IP(\mathcal{A}, w)$ we denote the set of all interactive processes in $\mathcal{A}$ with input w.

In order to represent an interactive process π, we also use the "arrow notation" for $\pi : D_0 \to^{a_1} D_1 \to^{a_2} \cdots \to^{a_n} D_n \to D_{n+1} \to \cdots$.

For an interactive process π in $\mathcal{A}$ with input w, if $En_A^p(D_m) = \emptyset$ for some $m \geq |w|$, then we have that $Res_A(D_m) = \{D_m\}$ and $D_m = D_{m+1} = \cdots$. In this case, considering the smallest m, we say that π *converges on* D_m (at the m-th step). When an interactive process π converges on D_m, each D_i of π is omitted for $i \geq m+1$.

Definition 5. Let $\mathcal{A} = (S, \Sigma, A, D_0, f)$ be an RA. The *language accepted by* $\mathcal{A}$, denoted by $L(\mathcal{A})$, is defined as follows:

$$L(\mathcal{A}) = \{w \in \Sigma^* \mid \pi \in IP(\mathcal{A}, w) \text{ that converges on } D_m \text{ at the } m\text{-th step, for some } m \geq |w|, \text{ and } f \subseteq D_m\}.$$

Example 1. Let us consider a reaction automaton $\mathcal{A} = (S, \Sigma, A, D_0, f)$ defined as follows:

$S = \{a, b, c, d, e, f\}$ with $\Sigma = \{a\}$,
$A = \{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4, \mathbf{a}_5, \mathbf{a}_6\}$, where
$\mathbf{a}_1 = (a^2, \emptyset, b)$, $\mathbf{a}_2 = (b^2, ac, c)$, $\mathbf{a}_3 = (c^2, b, b)$,
$\mathbf{a}_4 = (bd, ac, e)$, $\mathbf{a}_5 = (cd, b, e)$, $\mathbf{a}_6 = (e, abc, f)$,
$D_0 = d$.

Let $w = aaaaaaaa \in S^*$ be the input string and consider an interactive process π such that

$$\pi : d \to^a ad \to^a bd \to^a abd \to^a b^2d \to^a ab^2d \to^a b^3d \to^a ab^3d \to^a b^4d \to c^2d \to bd \to e \to f.$$

It can be easily seen that $\pi \in IP(\mathcal{A}, w)$ and $w \in L(\mathcal{A})$. For instance, since $\mathbf{a}_2^2 \in En^p_{\mathcal{A}}(b^4d)$, it holds that $c^2d \in Res_{\mathcal{A}}(b^4d)$. Hence, the step $b^4d \to c^2d$ is valid. We can also see that $L(\mathcal{A}) = \{a^{2^n} \mid n \geq 1\}$ which is context-sensitive (see Figure 1-(i)).

We shall show the equivalence of the accepting powers between reaction machines and Turing machines. For the details of proof, we refer [6].

Theorem 1. *Every recursively enumerable language is accepted by a reaction automaton.*

4 Space Complexity Classes

We now consider space complexity issues of reaction automata. That is, we introduce some subclasses of reaction automata and investigate the relationships between classes of languages accepted by those subclasses of automata and language classes in the Chomsky hierarchy.

Let $\mathcal{A}$ be an RA and f be a function defined on $\mathbf{N}$. Motivated by the notion of a workspace for a phrase-structure grammar ([8]), we define: for $w \in L(\mathcal{A})$ with $n = |w|$, and for π in $IP(\mathcal{A}, w)$,

$$WS(w, \pi) = \max_i\{|D_i| \mid D_i \text{ appears in } \pi\ \}.$$

Further, the *workspace of* $\mathcal{A}$ *for* w is defined as:

$$WS(w, \mathcal{A}) = \min_\pi\{WS(w, \pi) \mid \pi \in IP(\mathcal{A}, w), \text{ where } \pi \text{ converges.}\}.$$

Definition 6. (i). An RA $\mathcal{A}$ is $f(n)$*-bounded* if for any $w \in L(\mathcal{A})$ with $n = |w|$, $WS(w, \mathcal{A})$ is bounded by $f(n)$.
(ii). If a function $f(n)$ is a constant k (linear, polynomial, exponential), then $\mathcal{A}$ is termed k-bounded (resp. linearly-bounded, polynomially-bounded, exponentially-bounded), and denoted by k-RA (resp. *lin*-RA, *poly*-RA, *exp*-RA). Further, the class of languages accepted by k-RA (*lin*-RA, *poly*-RA, *exp*-RA, arbitrary RA) is denoted by k-$\mathcal{RA}$ (resp. $\mathcal{LRA}, \mathcal{PRA}, \mathcal{ERA}, \mathcal{RA}$).

Let us denote by $\mathcal{REG}$ ($\mathcal{LIN}, \mathcal{CF}, \mathcal{CS}, \mathcal{RE}$) the class of regular (resp. linear, context-free, context-sensitive, recursively enumerable) languages.

We show two characterizations concerning $\mathcal{LRA}$ and $\mathcal{ERA}$ in relation to the Chomsky hierarchy, and two interesting results. One is concerned with a representation theorem for the class $\mathcal{RE}$ in terms of $\mathcal{LRA}$, and the other is a new characterization of $\mathcal{CS}$ with $\mathcal{ERA}$ (for the proofs, see [7]).

Theorem 2. *For any recursively enumerable language L, there exists an LRA $\mathcal{A}$ such that $L = h(L(\mathcal{A}))$ for some projection h.*

Theorem 3. *The following inclusions hold :*
(1). $\mathcal{CS} = \mathcal{ERA}$.
(2). $\mathcal{REG} = k\text{-}\mathcal{RA} \subset \mathcal{LRA} \subseteq \mathcal{PRA} \subset \mathcal{ERA} \subset \mathcal{RA} = \mathcal{RE}$ (for each $k \geq 1$).
(3). *$\mathcal{LIN}$ ($\mathcal{CF}$) and $\mathcal{LRA}$ are incomparable.*

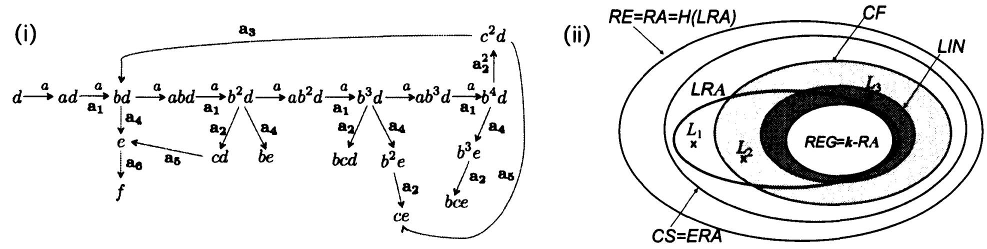

図 1: (i) Interactive processes for accepting a^2, a^4 and a^8 in $\mathcal{A}$. (ii) Language class relations in the Chomsky hierarchy : $L_1 = \{a^n b^n c^n \mid n \geq 0\}$; $L_2 = \{a^m b^m c^n d^n \mid m, n \geq 0\}$; $L_3 = \{ww^R \mid w \in \{a, b\}^*\}$.

5 Concluding Remarks

Based on the formal framework presented in a series of papers [3, 4, 5], we have introduced the notion of reaction automata and investigated the language accepting powers of the automata. Roughly, a reaction automaton may be characterized in terms of three key words as follows : a *language accepting device* based on the *multiset rewriting* in the *maximally parallel manner*. Specifically, we have shown that reaction automata can perform the Turing universal computation.

Moreover, we investigate reaction automata with a focus on the formal language theoretic properties of subclasses of reaction automata. We have shown (i) any recursively enumerable language can be expressed as a homomorphic image of a language in $\mathcal{LRA}$, (ii) the class $\mathcal{ERA}$ coincides with the class of context-sensitive languages.

Many subjects remain to be investigated along the research direction suggested by reaction automata in this paper. Most of all, it is of importance to explore the relationship between RAs and other computing devices that are based on the multiset rewriting, such as a variety of P-systems and their variants ([2]).

参考文献

[1] C. Calude, Gh. Păun, G. Rozenberg and A. Salomaa (Eds.), *Multiset Processing*, LNCS 2235, Springer, 2001.

[2] Gh. Paun, G. Rozenberg, A. Salomaa (Eds.), *Handbook of Membrane Computing*, Oxford University Press, 2010.

[3] A. Ehrenfeucht, G. Rozenberg, Reaction systems, *Fundamenta Informaticae* vol.75, pp.263-280, 2007.

[4] A. Ehrenfeucht, G. Rozenberg, Events and modules in reaction systems, *Theoretical Computer Science* vol.376, pp.3-16, 2007.

[5] A. Ehrenfeucht, G. Rozenberg, Introducing time in reaction systems, *Theoretical Computer Science* vol.410, pp.310-322, 2009.

[6] F. Okubo, S. Kobayashi, T. Yokomori, Reaction Automata, arXiv:1201.3082v2, to appear in *Theoretical Computer Science*.

[7] F. Okubo, S. Kobayashi, T. Yokomori, On the Properties of Language Classes Defined by Bounded Reaction Automata, arXiv:1111.5038v2.

[8] A. Salomaa, *Formal Languages*, Academic Press, New York, 1973.

[9] Y. Suzuki, Y. Fujiwara, J. Takabayashi, H. Tanaka, Artificial Life Applications of a Class of P Systems, in: *Multiset Processing*, C. Calude, Gh. Păun, G. Rozenberg, A. Salomaa (Eds.), LNCS 2235, Springer, pp.299-346, 2001.