Final Report on the Research Project

"Nature-motivated computational models in the theory of formal languages and automata",

supported by the Hungarian Research Fund, Grant. No. K 75952

1 The aim of the project

We intended to construct and develop nature-motivated computational models and study their properties, with the aim of enriching the theory of formal languages and automata and to provide tools for better understanding the theoretical principles, the driving forces behind natural processes. We focused on studying P automata (membrane automata) which combine the features of traditional automata and nature-motivated distributed architectures. Our aim was to study the effect of their architecture and the way of the organization of their functioning elements on the computing power, the descriptional complexity, the robustness and on the dynamical patterns of their behavior. We planned to develop a suitable classification of P automata classes which might contribute to the development of the concept of a "natural" automaton.

We also investigated the properties of networks of elementary processors based on simple formal-language-theoretic operations motivated by properties of DNA molecules or the genome (bio-inspired processors). These systems are interesting in exploring the limits of computing devices with very simple operations and architecture but large computing power.

2 The project, main achievements, participants

Our investigations followed the goals of the project, but they were also driven by the natural developments of our achievements, and the main trends and open problems in the corresponding scientific fields. The research and the obtained results correspond to the planned investigations, the work was performed in a competitive international scientific environment.

We obtained important results, especially in the theory of P automata and their variants, introduced promising new models and discovered connections between classical automata theory and P automata theory. We also opened research directions by demonstrating connections between P systems theory (membrane computing) and other scientific areas. Our results provide new knowledge on size, computational, and "functional" complexity of the considered constructs, and contribute to the development of membrane computing, formal languages, and automata theory. We performed the research in an international community, cooperating mainly with researchers from the area of membrane and molecular computing. Our achievements have obtained attention which is proved, among others, by our invited talks at leading international

workshops and conferences in membrane computing and formal language theory (WMC10: publications, item [6]; CMC12: publications, item [15]; CMC15: publications, item [28]; DLT 2012: publications, item [16]; NCMA 2013: publications, item [20]). The obtained results were published in proceedings of international conferences (LNCS, Springer), edited volumes (IOS Press) and international journals (Theoretical Computer Science, International Journal of Foundations of Computer Science). Four papers are in status "accepted" by the international journal Fundamenta Informaticae and one book chapter is in status "to appear" at a book published by Springer.

Both participants, Erzsébet Csuhaj-Varjú and György Vaszil, actively participated in the research. The project was suspended (due to our request) in the period 01.06.2012-31.05.2013, since both participants had to change their workplace (from a research institute to a university) and the significantly increased teaching load and the large amount of new type of tasks made the intensive work in the project impossible. In the meantime, both participants have become heads of departments, thus due to our request the final phase of the project was extended to enable the completion of some of the investigations. These changes also had some impact on the financial plan, we had to ask for some reasonable modifications.

3 Results

Our research focused on variants of P systems (membrane systems). These constructs are distributed biomolecular computing devices inspired by the architecture and the functioning of living cells. The main ingredient of a P system is a hierarchically embedded structure of membranes. Each membrane encloses a region that contains multisets of objects. There are rules describing the evolution of the objects present in the regions and their communication to other regions. The evolution of the system corresponds to a computation. P systems theory or membrane computing is a well-established, vigorous scientific area of natural computing.

4 P automata

P automata are important variants of purely communicating (symport/antiport) P systems which accept strings in an automaton-like fashion. Strings in the language of a P automaton are obtained as mappings of the multiset sequences which enter the P system during an accepting computation. In a basic article (Csuhaj-Varjú, Ibarra, Vaszil, 2004), the case of simple, non-erasing input mappings was examined. It was shown that if the rules of the P automaton are applied sequentially (one rule in each region at every computation step), then the accepted language class is strictly included in the class of languages accepted by nondeterministic one-way Turing machines with a logarithmically bounded workspace, more precisely, it equals a complexity class we called $rNLOG$, where

the workspace available during the computation depends not on the length of the whole input, but on the length of the part of the input which is already processed. If the rules are applied in the maximally parallel manner (in one step, as many rules are applied simultaneously as possible), then the class of context-sensitive languages is obtained. A widely studied type of input mappings is f_{perm}, which maps a multiset to the set of strings which consists of all permutations of its elements. It is a natural concept since it reflects the fact that the objects which appear simultaneously can be observed in any order. Although P automata with f_{perm} was intensively studied, its precise language accepting power has not been determined. We have shown that both in the sequential and in the maximally parallel rule application mode these P automata variants describe a class of languages properly included in the class of languages in $rNLOG$. Furthermore, we also proved that the number of membranes in P automata defined over unary alphabet and using f_{perm} in arbitrary working mode induces a strict infinite hierarchy of the corresponding language classes according to inclusion. To prove the results, we introduced two variants of counter machines, making it possible to read multisets (represented as sets of all permutations) and manipulating counters in a conventional manner (see publications, items [22], [30]).

We also proved that dP automata using f_{perm} as input mapping, i.e., distributed systems of P automata, are as powerful as the class of distributed systems of special counter machine acceptors (publications, items [23], [29]).

According to the generic variant, a string accepted by a dP automaton is defined as the concatenation of the strings accepted by the individual components during a computation performed by the system. We introduced two variants of languages based on agreement of the components (weak and strong agreement languages), where the components accept the same language. Using these notions, we described connections between dP automata and nondeterministic multi-head finite automata. We showed that languages of one-way nondeterministic multi-head finite automata correspond to agreement languages of finite dP automata (dP automata having a finite number of configurations). We also introduced the notion of a two-way finite dP automaton, with double alphabets (left-right move), and demonstrated that two-way finite dP automata characterize the language family accepted by nondeterministic two-way multi-head finite automata, via weak and strong agreement languages. The latter theorems provide characterization of the complexity class $NLOG$ (publications, items [10], [18]).

We have also obtained important results on the efficiency of dP automata. A language accepted by a P automaton is efficiently parallelizable if there exists a dP automaton which accepts the same language and its computations only take some fraction of the computing time of the non-distributed variant. Earlier it was shown that considering the class of mappings f_{perm}, there are efficiently parallelizable regular languages. In contrast to this, we have shown that the notion of efficient parallelizability depends on the input mapping used, in particular, we have demonstrated that considering the more general class of mappings which can be realized by finite transducers, there are no efficiently

parallelizable regular languages (with respect to this class). See publications, items [14],[15],[19].

The results in this section form a basis of a taxonomy of P and dP automata, since they demonstrate that the input mapping f defining the language and the different working modes are features that separate P and dP automata classes from each other. The results also expand tools of automata theory and provide information on the size and computational complexity of language classes defined by P and dP automata.

5 Generalized communicating P systems

One important problem area in membrane computing is to distinguish P systems with large computational power and at the same time with simple architecture, simple functioning rules, and small size complexity. An example for such devices is the generalized communicating P system (GCPS). It corresponds to a hypergraph where each node is represented by a cell and each edge is represented by a rule. Every cell contains a multiset of objects which - by communication rules - may move between the cells. Each communication rule, called also an interaction rule, moves two objects to new destination cells. Depending on their forms, several restrictions on communication rules (modulo symmetry) can be introduced; we distinguish split, join, chain, parallel-shift, etc rules. If the GCPS has only one type of rule, then we speak of a minimal interaction P system, a GMCPS, for short. Earlier it was shown that any register machine can be simulated by a GMCPS having nineteen cells and using only parallel-shift rules. We proved that in most of the remaining cases (in seven out of eight) the corresponding GMPCSs are able to determine any recursively enumerable set of non-negative integers, in the remaining case these systems determine finite singletons of natural numbers. The proofs demonstrate that the large expressive power can be obtained by systems with relatively small numbers of cells and simple graph architectures. The architectures of the systems, presented in the proofs, were built from "functional blocks of cells" which determined behavioral primitives, for example, incrementing or decrementing the number of objects at a given cell (publication list, items [4],[14]) We have also shown that minimal interaction P systems over a singleton alphabet given with any of the parallel-shift, presence-move, chain, and join rules are computationally complete, while in the case of conditional-uniport-in rules these constructs are less powerful than the previous ones (publications, item [7]).

We also introduced and studied generalized communicating P automata and proved that most of these constructs, even using only rules of type split, or symport2, or join, or conditional-uniport-in, are able to recognize any recursively enumerable language, while in the case of antiport1 rules they recognize the elements of the class of regular languages (list of accepted publications, item [1]).

6 P colonies

P colonies are very simple membrane systems, which consist of a collection of cells in a shared environment, each having the same number of objects inside and an associated set of programs (each program is a finite set of rules). Communication between the cells is only possible indirectly through the environment. The number of objects in a cell is constant during the computation.

P colonies have been studied in detail during the years and it has been shown that they, even with very small size properties, are able to identify any recursively enumerable set of non-negative integers. Completing previous investigations, we proved that P colonies with six components and one object in each cell are able to generate any recursively enumerable set of vectors even if no checking rule is used (checking rules are able to establish the presence of an object in the environment). We also examined special variants of P colonies with two objects, called P colonies with senders and consumers, having special rules for insertion-deletion. Those components which have only insertion rules are called senders, those which have only deletion rules are called consumers. We proved that for these P colony variants three cells, one sender and two consumers are sufficient to generate all recursively enumerable sets of integer vectors (publications, items [1],[2]).

To study the connection between P colony members and the environment, we introduced the notion of a P colony automaton (a PCol automaton). To describe the situation when the behavior of the components of the P colony is influenced by direct impulses coming from the environment step-by-step, the generic model is augmented with a string put on an input tape to be processed. In addition to the rewriting rules and the rules for communicating with the environment, the cells have so-called tape rules which are used for reading the next symbol on the input tape. PCol automata may work in several computation modes: some of them affect the tape, some affect only the environment. We have shown that in some of the working modes PCol automata are computationally complete, however, there are cases when the recognizing power is reduced to the class of regular languages (publications, item [5]).

We extended the notion of a PCol automaton to generalized PCol automaton, GenPCol automaton for short, which reads a multiset of several symbols in one computational step, instead of just one symbol. We have shown that if we consider the non-restricted functioning mode of GenPCol automata which does not require the automata to have tape rules in all of its programs, nor do we restrict its rules by considering only communication rules as tape rules, then GenPCol automata characterize the class of recursively enumerable languages, while in the other two cases they determine the class of languages computed in restricted logarithmic space ($rNLOG$, see above). We also demonstrated the close connection between P automata and generalized P colony automata (publications, items [31]).

We also examined P colonies as string processors, namely, we introduced and studied P colonies where the environment is given as a string. These constructs are called automaton-like P colonies or APCol systems. The interaction

between the agents in the P colony and the environment is realized by exchanging symbols between the objects of the agents and that of the environment (communication rules). We showed that the family of ε-free languages accepted by so-called jumping finite automata is properly included in the family of languages accepted by APCol systems with one agent and any ε-free recursively enumerable language can be obtained as a projection of a language accepted by an automaton-like P colony with two agents (publications, item [27]; list of accepted publications, item [3])

These results on P colonies contribute to finding the limits of syntactic simplicity of computationally complete computational devices. The reader may notice that these systems, especially the automata variants of P colonies can be considered as networks of very simple processors that manipulate a string (set of strings; multiset) with very simple operations which correspond to point mutations (insertion, deletion, replacement of a symbol). Thus, the above P colony variants can also be considered as bio-inspired language processors. Their close relation the so-called networks of evolutionary processors (networks of string processors using point mutations) can easily be demonstrated.

7 P systems with topologies

Membrane systems introduce in a very natural way a specific topology, where the membranes delimit regions (compartments) containing local objects and interaction rules, together with specific links between compartments. These links describe communication channels allowing adjacent compartments to exchange objects. Although this topology is flexible enough for modeling various natural or engineering systems, there are cases when a finer grain topological structure is required. Therefore, we investigated the use of a topological space as a framework to control the evolution of the system with respect to a family of open sets that is associated with each region. This approach produces a fine grain description of local operations occurring in each region or between adjacent regions by restricting the interactions between objects to those from a given neighborhood. This initial study demonstrated the influence of an arbitrary topology on the way basic membrane systems compute (publications, item [17]).)

Developing the theory further, we investigated the use of general topological spaces in connection with a generalized variant of membrane systems. We prescribed a restriction on the interaction of objects given by open sets of a topology and multisets of objects associated with them, which dynamically change during the functioning of the system and which together define a notion of vicinity for the objects taking part in the interactions. We showed how the application of the rules is defined and the computation process is affected in this context (list of accepted publications, item [4]).

8 P systems and other scientific areas

We have studied the connections between P systems and other scientific areas, disciplines as well.

We looked at the field of membrane computing as a particular example of the so-called chemical computational paradigm. This paradigm aims to describe computations in terms of a symbolic chemical solution of molecules and the reactions which can take place between them. Its origins goes back to the Gamma programming language (introduced by J.P. Banâtre and D. Le Métayer in the 1980's). We studied the relationship of membrane systems and logical formalisms based on the chemical paradigm, and showed how certain types of membrane system computations can be described in them (publications, items [24], [26]; list of accepted publications, item [2]). These results represent the first steps in the direction of establishing the relationship of the two paradigms, which could be interesting from several points of view. For example, by being able to translate chemical programs to membrane systems, we could obtain a high level programming language for the description of membrane algorithms.

We also introduced some variants of P systems that mimic the behaviour of social networks and illustrated some of the characteristics of them. In this case, the functioning of the P systems is governed by communication, which provides new aspects both in P systems theory and in formal aspects of social networks. We also discussed other concepts related to social networks and suggested suitable classes of P systems. were suggested (publications, item [11]). Based on our invited talk at conference CMC15 (publications, item [28]), an extended paper is in preparation.

We also demonstrated connections between P automata and complex systems (publications, item [20]). Complex systems are subject of a field of science studying how parts of a system give rise to the collective behaviors of the system, and how the system interacts with its environment. The study of complex systems is an interdisciplinary scientific field which focuses on questions about wholes, parts, and relationships. In this general sense, P automata can be considered as models for complex systems, due to their original motivation and some features models for natural systems as well. The objects in the membrane architecture can be considered as very simple, elementary components, parts or agents, collections of which interact with each other and with the environment of the system. The interactions defined by the rules are local and non-linear, the latter property means that small causes can have large results. The sequences of multisets moving across the skin membrane during the functioning of the P automaton and their maps under some mappings describe the behavior of the system.

9 Further Achievements

In addition to the results listed in the previous sections, we raised several further ideas in our invited talks and in the survey paper M. Gheorghe, Gh. Păun, M.-

J- Pérez-Jiménez, G. Rozenberg: Research Frontiers of Membrane Computing: Open Problems and Research Topics, International Journal of Foundations of Computer Science, Vol. 24, No. 5 (2013) 547623. This paper also contains a collection of contributions of other authors. Our sections are the following: E. Csuhaj-Varjú P colonies and P automata, pp. 562- 565, and Gy. Vaszil: Speeding up P automata, pp. 570-572.

We also summarized the achievements and raised some problems in membrane computing and in the theory of networks of evolutionary processors in our survey article (publications, item [9]).

10 Applicability of the results

Since our research is basic research, our results cannot be directly applied in practice. However, the application of the obtained results and new ideas as models, tools, or new approaches can be expected in other scientific fields (Section "P systems and other scientific fields") or in classical computing. In papers (publications, item [25]; list of accepted publications, item [5]) an interesting connection is established between P automata and the complexity of Turing machine computations. It is shown, that in the deterministic case, the class of languages described by restricted space bounded Turing machines coincides with the one corresponding to so-called strongly space bounded Turing machine computations (the notion of restricted space bound was introduced to describe the complexity of P automata computations, see $rNLOG$ above).

11 List of accepted publications

1. E. Csuhaj-Varjú, Gy. Vaszil: Generalized Communicating P Automata, In: A. Adamatzky (ed.): Automata, Universality, Computation, Tribute to Maurice Margenstern, Emergence, Complexity and Computation 12, Springer, pp., 219-236. DOI: 10.1007/978-3-319-09039-9 (to appear).

2. P. Battyányi, Gy. Vaszil: Describing Membrane Computations with a Chemical Calculus. Fundamenta Informaticae, accepted.

3. L. Cienciala, L. Ciencialová, E. Csuhaj-Varjú: P colonies processing strings. Fundamenta Informaticae, accepted.

4. E. Csuhaj-Varjú, M. Gheorghe, M. Stannett, Gy. Vaszil: Spatially localised P Systems. Fundamenta Informaticae, accepted.

5. M. Kutrib, J. Provillard, Gy. Vaszil, M. Wendlandt: Deterministic one-way Turing machines with sublinear space. Fundamenta Informaticae, accepted.