Communicating Mobile Nano-Machines and Their Computational Power*
(Extended abstract)

Jiří Wiedermann[1] and Lukáš Petrů[2]

[1] Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Prague 8, Czech Republic
jiri.wiedermann@cs.cas.cz

[2] Faculty of Mathematics and Physics, Charles University, Malostranské náměstí 25, 118 00 Prague 1
Czech Republic
lukas.petru@st.cuni.cz

Abstract. A computational model of molecularly communicating mobile nanomachines is defined. Nanomachines are modeled by a variant of finite-state automata—so-called timed probabilistic automata—augmented by a severely restricted communication mechanism capturing the main features of molecular communication. We show that for molecular communication among such motile machines an asynchronous stochastic protocol originally designed for wireless (radio) communication in so-called amorphous computers with static computational units can also be used. We design an algorithm that using the previous protocol, randomness and timing delays selects with a high probability a leader from among sets of anonymous candidates. This enables a probabilistic simulation of one of the simplest known model of a programmable computer—so-called counter automaton—proving that networks of mobile nanomachines possess universal computing power.

Keywords: molecular communication; nanomachines; timed probabilistic automata; communication protocol; universal computing.

1 Introduction

Nanomachines are molecular cell-sized artificial devices or engineered organisms produced by self-assembly or self-replication, capable of performing simple tasks such as actuation and sensing (cf. [4]). Construction of various nanomachines seems to be entirely within reach of current nano- and bio-technologies (cf. [3]). Once constructed and endowed with a certain sensor and actuating abilities communication among nanomachines becomes an important problem.

* This research was carried out within the institutional research plan AV0Z10300504 and partially supported by the GA ČR grant No. 1ET100300419 and GD201/05/H014.

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2009

Corresponding molecular mechanism design presents a great challenge for nanotechnology, bio-technology, and computer science. In molecular biology there is an explosively growing field dealing with molecular communication. Here, the respective research is interested almost exclusively in the (bio)chemical aspects of existing communication mechanisms in living systems. It seems that almost no attention has been paid to the algorithmic aspects of the respective communication process. Communication at a nanoscale substantially differs from communication scenarios and frameworks known from classical distributed systems. Key features of traditional versus molecular communication have been neatly summarized in [4]. In molecular communication, the communication carrier is a molecule; chemical signals are extremely slowly propagated by diffusion in an aqueous environment with low energy consumption. In general, it is not necessary that the signal will reach all targets: a majority will do. This is to be compared with traditional communication via electromagnetic waves where electronic or optical signals are propagated at light speed with high energy costs in an airborne medium and message delivery to all targets is required.

In what follows we will concentrate onto a scenario in which nanomachines form an autonomous system operating in a closed liquid environment without external control (a similar scenario is also considered in [4]). The system consists of a finite number of nanomachines freely "floating" in their environment that interact via molecular communication creating thus a kind of ad-hoc network. We assume that there is a sufficient number of nanomachines such that within the "communication radius" (to be explained later) of each machine there are other nanomachines available. Within the communication process the authors of [4] have identified following steps: encoding of information onto the information molecules, sending of the carrier/information molecules into the environment, propagation of the carrier/information molecules through the environment, receiving of the carrier/information molecules, and decoding of the information represented by the received information molecules into reaction at a receiver. In addition, recycling of carrier/information molecules may be necessary to avoid accumulation at a receiver.

This seems to be a reasonable sequencing of communication subtasks, but from an algorithmic viewpoint several questions immediately arise. What protocol is used for molecular communication? What happens if there are more nanomachines communicating concurrently? Would it be necessary to synchronize them so that one would act as a sender and the other as a receiver? How does the sending machine learn that a target machine has received its signal? If we allow a finite number of different signals (types of molecules) by which the machines can communicate, what happens if a machine having several receptors detects different signals at different receptors at the same time? What happens when a (would be) receiving machine is engaged in sending a signal? What are the computational limits of the underlying system?

It is the purpose of the present paper to answer such questions. In order to do so three things are needed: (i) a more detailed algorithmic (computational) model of a nanomachine, (ii) a communication protocol, i.e., an algorithm controlling the communication behavior of each nanomachine, and (iii) a simulation procedure showing the relation of networks of communicating nanomachines to a standard (classical) model of computations whose computational power is known.

The contribution of the paper corresponds to the latter mentioned three items. First, the paper defines a new model of communicating nanomachines based on probabilistic timed finite state automata. This seems to be a novel application of such types of automata. The size of the resulting network of nanomachines is scalable even though the number of states of each nanomachine remains fixed (i.e., independent of the number of communicating machines). The most important feature of our nanomachine model is its part respecting the limitations of molecular communication. The communication mechanism works with the minimal functionalities available at the molecular level: a finite number of states, randomness, asynchronicity, anonymity of nanomachines, and one-way communication without a possibility of signal reception acknowledgement. Second, the paper shows that for basic communication among the mobile nanomachines a protocol originally designed for wireless radio communication in the case of so-called amorphous computers with the static computational units can be used. This points to the robustness of the respective protocol that is used in a completely different communication medium and for mobile, rather than static communicating units. Last but not least, we show that nanomachines captured by our modelling are able to perform whatever kind of computation—they possess universal computational power.

The structure of this extended abstract also mirrors the previous three items. In Section 2 the nanomachine computational model is introduced and in Section 3 it is shown that wireless communication protocol designed in [7] can be also used for the case of molecular communication. In Section 4 a simulation of a very simple model of universal computer—so-called counter automaton—by nanomachines is shown.

The full version of the paper is available as a technical report [8].

2 Nanomachine Computational Model

From computational point of view we will see each nanomachine as a timed probabilistic finite-state automaton. In essence this is a finite state automaton augmented with quantitative information regarding the likelihood that transitions occur and the time at which they do so. Timing is controlled by a finite set of real valued clocks. The clocks can be reset to 0 (independently of each other) with the transitions of the automaton, and keep track of the time elapsed since the last reset. The transitions of the automaton put certain constraints on the clocks values: a transition may be taken only if the current values of the clocks satisfy the associated constraints (cf. [2], [5]).

A biomolecular realization of a probabilistic automaton has been described in [1]. As far as timing mechanisms of biological automata are concerned, in biology there is a vast body of research dealing with biological oscillators and clocks controlling various biological processes in living bodies and it is quite plausible that such mechanisms could also be considered in nanosystems.

Our version of probabilistic timed automaton will in fact be a probabilistic timed transducer (Mealy automaton) having a finite number of input ports (receptors) and output ports (emitters). The signal molecules will be represented by elements of automaton's finite working alphabet. The conditions under which the nanomachines can communicate are designed so as to make the underlying molecular mechanism as simple as possible while capturing the constraints imposed on molecular communication:

1. Each automaton is able to work in two modes: in the receiving mode, reading (in parallel) the symbols (molecules) from its input ports, and in the sending mode, writing the same symbols to its output ports. A read operation is successful if and only if all symbols at all input ports are identical. Otherwise, when the symbols at the input ports differ, a so-called *communication collision* occurs: the read operation fails and the symbols are released from the input ports. Except of communication ports, a nanomachine can also have receptors detecting other than signaling molecule stimuli, and other actuators doing some job corresponding to the purpose to which the nanomachine has been designed.
2. In the sending mode, an automaton releases signal molecules of the same kind through all its output ports; these molecules diffuse in the environment and eventually can reach the input ports of an automaton in a receiving mode.
3. After a certain time, signal molecules disintegrate into other molecules which are not interpreted by the automata as signal molecules. In their life time, signal molecules can travel, by diffusion, in average a certain maximal distance called *communication radius.*
4. The automata work asynchronously — there is no global clock in the system. The actions of each automaton are governed by automaton's local clock. The local clocks in the automata are not synchronized, however, they all "tick" (roughly) at the same rate since they all are realized by the same biochemical oscillators. A slight variation (that in practice may also be caused, e.g., by temperature variations) in the clock rate does not harm our purposes.
5. The automata have no identifiers, i.e., for communication purposes all senders and receivers "look the same".
6. The automata are equipped by a finite set of timers (clocks) that are assigned to certain transitions. These timed transitions work as described in the beginning of the section.

Note that the previous conditions are quite restrictive. Condition 1 means that an automaton cannot simultaneously be in a sending and receiving mode; a signalling molecule reaching an input port of an automaton in a sending mode will not be detected by that automaton by that time. Condition 2 essentially says that if a broadcast from one automaton is "jammed" by a broadcast of an other automaton broadcasting different signal, then the receiving automaton does not accept any signal. Condition 3 ensures that signal molecules cannot "roam" for ever in the environment and that current signals eventually prevail over the old ones in the communication radii of the automata. Condition 4 says that we cannot count on automata switching their sending and receiving modes synchronously. Also note that our automata can move in their environment, either passively, due to the external forces (e.g., in a bloodstream), or actively, like some bacteria. This by itself, but the more so in conjunction with condition 5, prevents whatever kind of "acknowledgements" of received messages. Last but not least, observe that making use of a finite number of states (independent of the number of communicating automata) does not allow "storing" automata "addresses" (names) in automaton's states since the number of such addresses grow unboundedly with the number of nanomachines.

A multiset of communicating identical nanomachines is called a *nanonet.*

3 Communication Protocol

Thanks to a similarity between the computational model of "static" amorphous computer from [7] and the present model a similar communication protocol as in the latter case applies. In order to enable an algorithmic insight into the functionality of the protocol we briefly describe the necessary background.

Consider the so-called *communication graph* G of a given nanonet whose nodes are nanomachines (automata) and edges connect those automata which are within the communication radius of each other. The size of G will be measured in the number n of its nodes. Obviously, the topology of G depends on time since in general our automata move. In order to enable communication among all (or at least: a majority of) available automata in a nanonet most of the time G must have certain desirable properties. What we need are connected graphs with small diameter and a reasonable node degree. The most important property of G is its connectivity: connectivity is a necessary condition in order to be able to harness all processors. Graph diameter $D = D(n)$ bounds the length of the longest communication path. Finally, the node degree Q (i.e., the maximal neighborhood size of a node) determines the collision probability of signals.

In order to our communication algorithms work in the way we assume we will consider the families of so-called *well-behaved nanonets.* For any n, these are the nanonets whose underlying computational graph of size n stays connected and whose diameter and maximal neighborhood size stay bounded by $D = D(n)$ and a constant Q, respectively, all the time.

The requirements put on the well-behaved nanonets are quite strong and one can hardly imagine that in "practice" they will be fulfilled, indeed. Nevertheless, the invariance of these properties is needed in order to be able to analyze the correctness and efficiency of the communication primitives we will deal with in a sequel. After presenting these primitives we will see that they are sufficiently "robust" in order to operate correctly and with a similar efficiency also in instances of nanonets that occasionally, for short time, deviate from their well-behaved properties.

Protocol Send. For delivering a signal s from a node X to any node Y in its communication neighborhood with a given probability $\varepsilon > 0$ of failure a wireless *Protocol Send* designed in [7] is used.

The idea is for each node to broadcast sporadically, minimizing thus a communication collision probability in one node's neighborhood. This is realized as follows. Let T be time to transfer a signal between any two neighbors at the communication radius distance. Each node has a timer measuring *timeslots* (intervals) of length $2T$. During its own timeslot, each node is allowed either to listen till the end of its timeslot, or to send a signal at the very beginning of its timeslot and then listen till the end of this timeslot. At the start of each timeslot a node sends s with probability $p = 1/(Q+1)$ and this is repeated for $k = O(Q \log(1/\varepsilon))$ subsequent timeslots. The probability of a node sending s is controlled by the transition probability of the respective probabilistic automaton.

Assuming that all nodes send their signals asynchronously according to *Protocol Send,* in [7] it is shown that s will be received by Y in time $O(Q \log(1/\varepsilon))$ with probability at least $1 - \varepsilon$.

In order to allow the sending automaton to send s $k = k(\varepsilon)$ times in a row as required by the protocol its timer must be set to the interval $\approx kT$. Note that this is the time for which a node must be in the sending mode.

Algorithm Broadcast. In order to send a signal from a node to any other node of a nanonet which is not in the communication radius of the sending node the idea of flooding the network by that signal is used.

The main idea of *Algorithm Broadcast* is to use every node reached by a given signal s as a "retranslation station" that distributes s using *Protocol Send* further through the network. After retransmitting s each node stops retransmitting s — it locks itself with respect to s. However, any signal different from s is again retransmitted and afterwards the node locks itself again with respect to that last signal, etc. Thus, a locked node remains in the receiving mode until it gets unlocked by a different signal.

Again, in [7] is is shown that given any $\varepsilon : 0 < \varepsilon < 1$, Algorithm Broadcast delivers s to each node of $\mathcal{N}$ that has not been locked with respect to s in time $O(DQ \log(n/\varepsilon))$ and with probability $1 - \varepsilon$. Afterwards, all nodes in $\mathcal{N}$ will be in a locked state with respect to s.

In order to achieve the failure probability ε of the broadcasting algorithm *Protocol Send* must be performed with error probability ε/n. This calls for repeating each send operation in a node $k = O(Q \log(n/\varepsilon))$ times—i.e., this time k should grow not only with a decreasing ε, but also with increasing size of the nanonet. Similarly as in the case of *Protocol Send,* the respective timer is realized by means of the timing mechanism of the underlying automata.

Now it is time to return to the problem of non well-behaved nanonets. From the description of the communication protocol it is seen that occasional short-time connectivity "interruptions" could not harm the communication as long as constant k controlling the repeated sending trials is set high enough to overcome the interruption periods. A sufficiently high value of k will also help to overcome differences and drifts in the clock rates of the individual nodes, the switching times between sending and receiving modes, and occasional local increase of maximal neighborhood size which can block efficient communication. Finally, a generous estimate of the nanonet diameter (which is always bounded by n)

that is in fact determined by the shape of the closed environment in which the nanomachines operate will help to overcome the time variations of instantaneous nanonet diameter.

4 Distributed Computing through Nanomachines

Let us assume that each nanomachine gets its “own” input from the domain $\{0, 1\}$ through its sensors; this input can be used in the subsequent “collective” data processing. If outputs from the nanomachines are also restricted to the domain $\{0, 1\}$ then the respective nanonet can be seen as a device computing functions with their inputs and outputs represented in unary. Further assume that within the net there is a distinguished nanomachine called the *base*. In the full version of the paper [8] it is shown that under this scenario a nanonet can simulate a very simple model of a universal (i.e., programmable) machine called *counter machine* (cf. [6]), with high probability. A counter machine computes with the help of a finite number of so-called counters. These counters represent any positive integer number and the only allowable operation over counters is their testing for zero, and adding or subtracting a one (the later operations can only be used for counters representing a positive number).

In a simulating nanonet each counter storing a number $n \geq 0$ is represented by a set of nanomachines of cardinality exactly n whose states belong to a specific subset of states. To simplify matters, assume that all machines in a counter are in the same state, q, let us say. Testing such a counter for zero is easy: using *Algorithm Broadcast* the base station issues a “query” whether there is a nanomachine in state q. Using the same algorithm each machine in state q sends the answer “yes” which eventually, in time that can be computed from the estimates given in Section 3, will reach the base station. If no answer arrives in the given time, than the counter is zero.

In order to add a one to a counter represented by a set S of nanomachines in state q we must select a single machine that is not in state q to be added to S by changing the state of that nanomachine to q. To subtract a one from a counter represented by S we must select a single machine in S and change its state to a state different from q. In both cases, a single machine has to be selected from a set of machines. This operation is called leader selection. The *leader* of a set of nanonmachines is a single nanomachine which originally has belonged to that set but subsequently is put into a distinguished state that is different from the states of all other nanomachines in that set.

The algorithm for leader selection from among the candidate set of nanomachines works in rounds. The idea of the algorithm is as follows. Using their probabilistic mechanism, in each round the nanomachines from the candidate set throw a random coin giving output 0 or 1. If this will split the set of candidate machines into two non-empty subsets we proceed recursively with either subset. Otherwise we choose the non-empty set which with a high probability is a singleton set—the leader. The instructions “what to do” are sent by the base station using *Algorithm Broadcast.* For the details and complexity analysis, see the full paper [8]. The resulting simulation algorithm is unbelievably cumbersome since all the computation is performed in a unary counting system and the operations have to be repeated many times in order to reach a prescribed level of reliability. In [8] the following theorem is shown:

Theorem 1. *Let $\mathcal{M}$ be a counter machine with input of size n operating with a constant number of registers of size $O(n)$ in time $p(n)$. Then for any $\varepsilon : 0 < \varepsilon < 1$ there exist constants $c_1 > 0$ and $c_2 > 0$ and a nanonet $\mathcal{N}$ consisting of $O(n)$ nanomachines using* Protocol Send *with failure probability ε such that $\mathcal{N}$ simulates $\mathcal{M}$ for inputs of size at most n in expected time $O(DQp(n)\log n \log(n/\varepsilon))$ with probability of failure bounded by* $\max\{1, c_2 p(n)(\varepsilon \log n + (\varepsilon/n)^{c_1})\}$.

Thus, the theorem claims that a nanonet of a fixed size can correctly, with a high probability, simulate any computation over inputs up to a certain size. For larger inputs a larger nanonet must be used. Most probably, in practice nobody would consider performing a universal computation by nanonets. Nevertheless, our result shows that in principle a nanonet can perform whatever algorithmic (or nano-robotic, if the machines embodiment is considered) task arising in practical applications.

References

1. Adar, R., Benenson, Y., Linshiz, G., Rozner, A., Tishby, N., Shapiro, E.: Stochastic computing with biomolecular automata. Proc. Natl. Acad. Sci. USA 101, 9960–9965 (2004)
2. Alur, R., Dill, D.L.: A Theory of Timed Automata. Theoretical Computer Science 126, 183–235 (1994)
3. Bath, J., Turberfield, A.J.: DNA nanomachines. Nature nanotechnology 2, 275–284 (2007)
4. Hiyama, S., Moritani, Y., Suda, T., Egashira, R., Enomoto, A., Moore, M., Nakano, T.: Molecular Communication. In: Proc. of the 2005 NSTI Nanotechnology Conference (2005)
5. Kwiatkowska, M., Norman, G., Parker, D., Sproston, J.: Performance Analysis of Probabilistic Timed Automata using Digital Clocks. In: Formal Methods in System Design, vol. 29, pp. 37–78. Springer, Heidelberg (2006)
6. Minsky, M.: Computation: Finite and Infinite Machines. Prentice-Hall, Englewood Cliffs (1967)
7. Wiedermann, J., Petrů, L.: On the Universal Computing Power of Amorphous Computing Systems. Published online in Theory of Computing Systems, January 27. Springer, New York (2009)
8. Wiedermann, J., Petrů, L.: Communicating Mobile Nano-Machines and Their Computational Power. Technical Report V-1024, Institute of Computer Science, Prague, May 2008, accessible from the web pages of the institute