How Does a P System Sound?

Manuel García-Quismondo, Miguel A. Gutiérrez-Naranjo,
Daniel Ramírez-Martínez

Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
mangarfer2@alum.us.es, magutier@us.es,
thebluebishop@gmail.com

University of Sevilla. Avda. Reina Mercedes s/n, 41012, Sevilla, Spain

Summary. P systems are computational devices versatile enough to represent many real-life scenarios. In this paper, we present a first interpretation for P systems where a computation produces a set of sounds. The idea is to associate sounds to the application of specific rules in the P system. The application of such rules produces sounds of one time unit. Different rules produce different sounds. The combination of such sounds along time can be interpreted as *music*.

1 Introduction

Membrane Computing is a nice symbolic game inspired by Nature. It abstracts the processes taking place in the compartmental structure of a living cell, interpreting them as computing operations. The information is usually represented as multisets of metabolites placed in compartments (membranes). According to different types of rules, the chemical compounds can evolve or be sent to other membranes. Depending on the model, the membranes can be created, divided or dissolved. These cellular devices are called P systems.

The description of a P system is usually made by a n-tuple which enumerates the different sets of components. Such sets usually involve alphabets for the symbols and labels, the multisets placed initially in the different regions and rules. Depending on the model, the description can also involve an alphabet for describing electrical charges or an explicit description of the membrane structure.

The aim of P systems is to perform changes on the multisets placed on the membranes (and, eventually, changes in the membrane structure) according to the set of rules. The rules are usually applied in a synchronous maximally parallel manner. It means that we can consider a clock which measures the time, taking the application of all the rules exactly one time unit. Rules are applied in parallel in a double way: rules are applied in all the membranes simultaneously and inside each membrane, all the objects that can evolve according to the rules must do it.

A configuration is an instantaneous description of a P system. After one time unit, several rules have been applied and the configuration has changed, so we obtain a new configuration. If no rules can be applied, we have a *halting configuration* and the computation stops.

In order to perform such changes, we need a *syntax* expressive enough to represent the different configurations of the P system, but we also need a *semantics* which allows us to apply the rules in an appropriate way. Changes in the syntax and/or the semantics produce the large panoply of P system models.

By considering the syntax and the semantics of a P system we can calculate the configurations which are reachable from a given configuration. Due to the inherent non-determinism of P systems, if one object can trigger more than one rule, one of them is chosen and the computation goes on. If we consider all the configurations to be reachable from a given one in each step, we obtain the so-called computation tree. In this abstract level, we can study many interesting problems related to confluence, reachability of configurations or halting criteria; problems inherent to all computation models.

Nonetheless, not only are we usually interested in the intrinsic properties of P systems themselves, but in using them as tools for dealing with other processes as well. Such processes can be of a symbolic nature, as solving the SAT [7] problem, or real-world problems such as the simulation of a population of bearded vultures [2] or the study of the epidermal growth factor receptor [8] in cells.

The design of P systems for dealing with these problems needs the explicit description of the syntax and the semantics of the problem, but we also need an *interpretation* of the symbols in our problem. According to the problem, the occurrence of a symbol in a membrane can be interpreted as a scavenger bird [1] in an ecosystem or a metabolite in the vesicle of a cell [4], but it also can represent a length unit of the width of the truck of a tree [9] or the negation of a literal in a propositional formula [7].

Other elements of the description of a P system can also different interpretations according to the problem which describes. In many situations, the electrical charges of the membranes act as *traffic lights*: a generated object can be several unit times *waiting* for a change in the polarization of a membrane which allows it to cross the membrane. Another typical example is to use a sequence of objects $\{z_1, \ldots, z_n\}$ as a *counter* till the appearance of the objects z_n which produces an event (maybe the dissolution or division of the membrane) exactly after n steps.

In this paper we provide a new interpretation to the application of a rule. We propose the use of membranes to produce sounds and, in a broad sense of the word, to produce music. The use of membranes to produce music should not be something surprising: from the origin of the Mankind, the vibration of tight membranes has been a source of sound. Nonetheless, to the best of our knowledge, this is the first time in which P systems are seen as *musical instruments*.

In the literature, there exist several approaches between music and computational biology, as [5] or [6] among others, but this is the first time in which P systems and music are brought together.

The paper is organized as follows: First we give some ideas about the interpretation of some computational events as sound producers. Next we provide some details about the implementation used in our experiments. In Section 4 we show an example of a P system designed for producing sounds and we finish with some final remarks.

2 Generating sounds

The word *sound* has two main meanings. On the one hand, it can be defined as the vibration transmitted through an elastic media (solid, liquid or gas), with frequencies in the approximate range of 20 to 20,000 hertz. The second meaning is related to the sensation stimulated in the organs of hearing by such vibrations in the air or other medium. Figure 1 shows the notes that can be usually reproduced by human voice or instruments.

	Oc. 0	Oc. 1	Oc. 2	Oc. 3	Oc. 4	Oc. 5	Oc. 6	Oc. 7	Oc. 8
C		32,70	65,41	130,81	261,63	523,25	1046,50	2093,00	4186,01
C#		34,65	69,30	138,59	277,18	554,37	1108,73	2217,46	
D		36,71	73,42	146,83	293,66	587,33	1174,66	2349,32	
D#		38,89	77,78	155,56	311,13	622,25	1244,51	2489,02	
E		41,20	82,41	164,81	329,63	659,26	1318,51	2637,02	
F		43,65	87,31	174,61	349,23	698,46	1396,91	2793,83	
F#		46,25	92,50	185,00	369,99	739,99	1479,98	2959,96	
G		49,00	98,00	196,00	392,00	783,99	1567,98	3135,96	
G#		51,91	103,83	207,65	415,30	830,61	1661,22	3322,44	
A	27,50	55,00	110,00	220,00	440,00	880,00	1760,00	3520,00	
A#	29,14	58,27	116,54	233,08	466,16	932,33	1864,66	3729,31	
B	30,87	61,74	123,47	246,94	493,88	987,77	1975,53	3951,07	

Fig. 1. Frequency in hertzs of the notes

In this paper we explore the relationship between P systems and music. In a broad sense, playing music consists on the production of vibration of a media (sounds) during several time units. The appropriate combination of the frequencies of the vibration (different notes) along time produces (eventually) a harmonious sound.

From this point of view, producing sounds with P systems is based on three key ideas:

- As we have seen above, time is generally considered to be discrete in P systems. We usually consider an universal clock for all the events (evolution of objects, creation, division or dissolution of membranes . . .). Such a universal clock can be considered as a uniform pulse. In other words, if we want to see a P system as a sound generator, we already have a metronome.

- We can associate a *sound* to an event produced in a P system in a time unit. The association is arbitrary and can be seen as an *interpretation*. In the same way as we can associate a *meaning* to a membrane, an object or to an electrical charge and see them as a geographical region, as a bearded vulture or as the signal of a traffic light, we can also associate a sound to an event produced during the computation.
  The main difference is that we associate a sound to the *application* of the rule, not to the rule itself. In such a way, the application of the rule is an event which is produced in one step of computation *at time t*. It makes sense to associate a sound to the application of a rule. The duration of the sound will be also arbitrarily prefixed. We will consider that the application of a rule produces one sound during a *time unit*.
- The third key point is the parallelism inherent in most of the P system models. As pointed above, in a time unit, several rules can be applied in different membranes. If we associate a sound to the application of one rule, we will produce several sounds *simultaneously*. In other words, for a musical piece, the application of some rules can produce the main voice and simultaneously, an accompanying *chord* can be performed. The parallelism of P systems opens a door to explore the possibilities of P systems as tools for producing polyphony.

3 Implementation

A first implementation of these ideas has been carried out by using the P system simulator SCPS [3]. This simulator was written in Prolog and one of its features is that the user can obtain a file which includes a report with the applied rules in each time unit.

3.1 Log parsing

These files can be easily parsed by using regular expressions or any other text search method, so a new *output format* can be defined in order to make the sound generation easier. Just to test this, three different musical interpretations can done by processing the *events*.

- **First method:** A different frequency is assigned to each membrane in the system. In this method, we consider that the membrane *vibrates* at the associated frequency when an object cross it, regardless of the object.
- **Second method:** A different frequency is assigned to each element. In this method, the vibration of one membrane does not depend on the membrane itself, but on the object which crosses the membrane.
- **Third method:** A different frequency is assigned to each rule. In this case, the frequency is associated to a membrane and an object, but it can be considered in a more general case, since other rules different from communication ones (dividing, dissolving, evolving, . . .) can also have an associated frequency.

3.2 Frequency and timing assignment

Time length for each musical event from a P system is an arbitrary choice as long as the human ear can hear each separate note. If the length of an event is too short, the output wave will be a valid physical recreation of the sonic wave, in terms of frequencies and amplitudes, but from an human point of view, it will be just a burst of beeps with no musical sense at all. Typical assignments for the first generated waves have been in the order of 0.1 seconds to 1 second. Greater lengths are, for now, just boring.

The frequency assignment for the events is, in a cold way, an arbitrary task too. Anyway, as far as somebody has to hear and to give an interpretation to the sounds, it seems to be quite obvious that a *friendly* assignment must be done. These assignments can be:

- **Harmonic frequencies:** Starting from an arbitrary frequency, the next ones are selected so they are integer multiples of that first one. So, if 440 Hz. is selected (central A), the next frequencies must be 880, 1320, 1760, . . . and so on. Integer dividers are allowed too.
- **Occidental dodecaphonic notes:** As it has been described above, each musical note in the Western culture is defined by a well-known frequency. If each, or some, of these frequencies are assigned to different events, any known chromatic melody can be played by the P system.
- **Occidental musical scales:** Well defined and largely used in the musical world, musical scales are subsets of the dodecaphonic system. In a scale, only some notes are picked from the entire the set, following any rule (typically, a set of intervals from a starting note). This way the scales of *C minor*, *B flat major*, *F diminished* or any other one, can be defined.

  Depending on what target to be reached, different assignments can be done.

- For *Musical P systems*, those which have been specifically designed to be interpreted as music event generators, all chromatic range or any scale will be a good choice, depending on the target melody.
- For *Non Musical P systems*, it seems to be a good idea to avoid reinforcing any possible inherent cacophony in the interpretation. As far as it is completely unknown how a P system sounds (until now), there is nothing such as a *correct assignment*, so harmonic frequencies or any scale notes are assigned. This way, at least, a *tuned* melody will be generated regardless of its musical coherence.

3.3 Wave generation

The process to generate a wave which represents how a P system sounds involves these steps:

1. Parse report (log) file from SCPS.
2. Select an interpretation (scales, harmonics,. . .).
3. Select parameters (time lengths, overlapping behavior, intensity levels,. . .).

4. Render the wave according to interpretation and parameters.
5. Post-processing the wave so as to make it *easy listened.*

The parsing is done by using regular expressions, which are a widely known tool for text search. The interpretation and the parameters are defined for harmonic and chromatic sounds inside the scripts as variable values. The render is accomplished following these steps:

1. Total length is calculated.
2. For each simulation step, every applied rule generates a single tone wave.
3. These single event waves are summed and the resulting wave can be normalized in amplitude. This exploits the parallelism, so chords are the interpretation of the set of rules which are fired in a step.
4. After every summed waves are linked together, the final wave can be normalized.
5. The resulting wave is saved as a PCM file with a sampling frequency of 44.100 Hz. and 16 bits, ready to be played in any standard WAV or music player.

4 Experimental Results

The first experiment performed was to consider a P system designed for an specific task without any relationship to music. The chosen one was a family of P system which solves the Subset Sum problem. We chose an instance of the problem, provided the corresponding input to the P system and got the information of the applied rules by using the simulator.

The obtained report file was processed by associating arbitrary sounds to the application of rules. The performance of this piece was presented during the Eight Brainstorming Week on Membrane Computing, held in Seville in the first days of February of 2010. The obtained sound was absolutely cacophonous, but it was the first answer to the question that entitles this paper.

The next challenge was to design a P system whose unique target was to produce sounds in a harmonious way. This first design is showed below. It was called Π_{HB}. From a technical point of view, it is a very simple P system. It only uses two types of rules:

- *Chemical decomposition or analysis:* $[\, a \to b\, c\,]_e$ The object a inside the membrane with label e is decomposed into two new objects b and c.
- *Osmotic reaction:* $[\, a\,]_e \to a\,[\,]_e$ The object a is sent out from the partially-permeable membrane with label e.

This first design Π_{HB} does not exploit the parallelism of P systems. In fact it is a deterministic P system which works sequentially, but illustrates the possibilities of P systems for producing sounds. In a more complex design, other membranes or objects could also perform parallel calculus and hence, producing the corresponding chords.

Let us consider now the following P system $\Pi_{HB} = (\Gamma, H, \mu, E, w_1, w_2, R)$ where

- $\Gamma = \{g_3, a_3, b_3, c_4, d_4, e_4, f_4, g_4, A_1, A_2, B_1, \ldots, B_4, C_1, \ldots, C_{11}, D_1, D_2, E_1, \ldots, E_4, F_1, F_2, G_1, \ldots, G_6, T_1, z_1, \ldots, z_{26}\}$ is the working alphabet.
- $H = \{1, 2\}$ is the set of labels.
- $\mu = [\,[\,]_2\,]_1$ is the membrane structure.
- $w_1 = \emptyset$ and $w_2 = \{g_3\, G_1\}$ are the initial multisets.
- R is the set of rules:

$$\begin{array}{llll} [g_3]_2 \to g_3\,[\,]_2 & [a_3]_2 \to a_3\,[\,]_2 & [b_3]_2 \to b_3\,[\,]_2 & [c_4]_2 \to c_4\,[\,]_2 \\ [d_4]_2 \to d_4\,[\,]_2 & [e_4]_2 \to e_4\,[\,]_2 & [f_4]_2 \to f_4\,[\,]_2 & [g_4]_2 \to g_4\,[\,]_2 \end{array}$$

$$\begin{array}{llll} [A_i \to a_3\,A_{i+1}]_2 & i \in \{1,3,5\} & [B_i \to b_3\,B_{i+1}]_2 & i \in \{1,2,3,5\} \\ [A_2 \to a_3\,G_2]_2 & & [B_4 \to b_3\,G_4]_2 & \\ [A_4 \to a_3\,G_6]_2 & & [B_6 \to b_3\,A_5]_2 & \\ [A_6 \to a_3\,F_1]_2 & & & \end{array}$$

$$\begin{array}{llll} [C_i \to c_4\,C_{i+1}]_2 & i \in \{1,3,4,5,7, 9,11,12,13\} & & \\ [C_2 \to c_4\,B_1]_2 & & [D_i \to d_4\,D_{i+1}]_2 & i \in \{1,3\} \\ [C_6 \to c_4\,G_8]_2 & & [D_2 \to d_4\,C_3]_2 & \\ [C_8 \to c_4\,B_5]_2 & & [D_4 \to d_4\,C_{11}]_2 & \\ [C_{10} \to c_4\,D_3]_2 & & & \\ [C_{14} \to c_4]_2 & & & \end{array}$$

$$\begin{array}{llll} [E_i \to e_4\,E_{i+1}]_2 & i \in \{1,3\} & [F_1 \to f_4\,F_2]_2 & \\ [E_2 \to e_4\,C_7]_2 & & [F_2 \to f_4\,E_3]_2 & \\ [E_4 \to e_4\,C_9]_2 & & & \end{array}$$

$$\begin{array}{ll} [G_1 \to g_3\,A_1]_2 & \\ [G_i \to g_3\,G_{i+1}]_2 & i \in \{2,4,6,8,9\} \\ [G_3 \to g_3\,C_1]_2 & \\ [G_5 \to g_3\,A_3]_2 & \\ [G_7 \to g_3\,D_1]_2 & \\ [G_{10} \to g_4\,G_{11}]_2 & \\ [G_{11} \to g_4\,E_1]_2 & \end{array}$$

4.1 Overview of the computation

The initial configuration consists on two membranes: An outer membrane with label 2 and an inner membrane with label 1. At the beginning, the membrane with label 1 is empty ($w_1 = \emptyset$) and the membrane with label 2 only has two objects: g_3 and G_1 ($w_2 = \{g_3\,G_1\}$), so the initial configuration can be expressed as $C_0 \equiv [\,[\,g_3\,G_1\,]_2\,]_1$.

From this initial configuration two rules can be applied: $[g_3]_2 \rightarrow g_3\,[\,]_2$ and $[G_1 \rightarrow g_3\,A_1]_2$. Both are applied simultaneously in one time unit. The object g_3 is sent out from membrane 2 and goes to membrane 1 and object G_1 is decomposed into objects g_3 and A_1. The new configuration at time $t = 1$ is $C_1 \equiv [\,[\,g_3\,A_1\,]_2\,g_3\,]_1$. Object g_3 in membrane 1 does not evolve any more, since there is no rule for it in the membrane 1. Nonetheless, objects in membrane 2 evolve according to the rules $[g_3]_2 \rightarrow g_3\,[\,]_2$ and $[A_1 \rightarrow a_3\,A_2]_2$. We obtain $C_2 \equiv [\,[\,a_3\,A_2\,]_2\,g_3^2\,]_1$, i.e., two copies of the object g_3 are placed in membrane 1 and objects a_3 and A_2 are placed in membrane 2. Bearing in mind that objects in membrane 2 do not trigger any rule, the remaining configurations can be obtained by applying simultaneously a chemical rule and an osmotic rule. Membrane 2 in configuration 46 contains objects c_4 and C_{14}. Rules $[c_4]_2 \rightarrow c_4\,[\,]_2$ and $[C_{14} \rightarrow c_4]_2$ are then applied and membrane 2 has only object c_4 in the configuration 47. Again $[c_4]_2 \rightarrow c_4\,[\,]_2$ is applied and finally, in configuration 48, membrane 2 is empty. No more rules can be applied, and the computation stops.

4.2 Interpretation

In order to obtain a melody, we associate a sound to an *event*. In this case, we choose to associate a note to the application of one *osmotic rule*. In this way, if the rule $[a]_2 \rightarrow a\,[\,]_2$ is applied at time t, then one sound will be produced during one time unit starting at time t. The remaining rules do not produce sounds. In order to simplify the interpretation, the objects sent out by osmotic rules have a natural interpretation in music. Such objects are g_3, a_3, b_3, c_4, d_4, e_4, f_4 and g_4. They are naturally interpreted by considering c_4 as the central note C in a keyboard piano. According to the computation, the first objects which cross out membrane e by application of the corresponding osmotic rules are g_3, g_3, a_3, etc. Figure 2 shows the melody obtained by the computation where the considered time unit corresponds to one quaver[1]

5 Final Remarks

In this paper we have presented a first approach between P systems and music. The idea of associating a sound to an event is quite natural and opens a large panoply of possibilities. The first one is to explore the possibilities of polyphony, since the parallelism is inherent to Membrane Computing devices. Other possibilities can be to introduce different timbres (different instruments) or different lengths of notes. Another exciting way to explore is the non determinism of P systems, since it can produce different melodies depending on the chosen computation.

[1] The G-clef and the time signature have been added. They do not correspond to the interpretation of the P system computation.

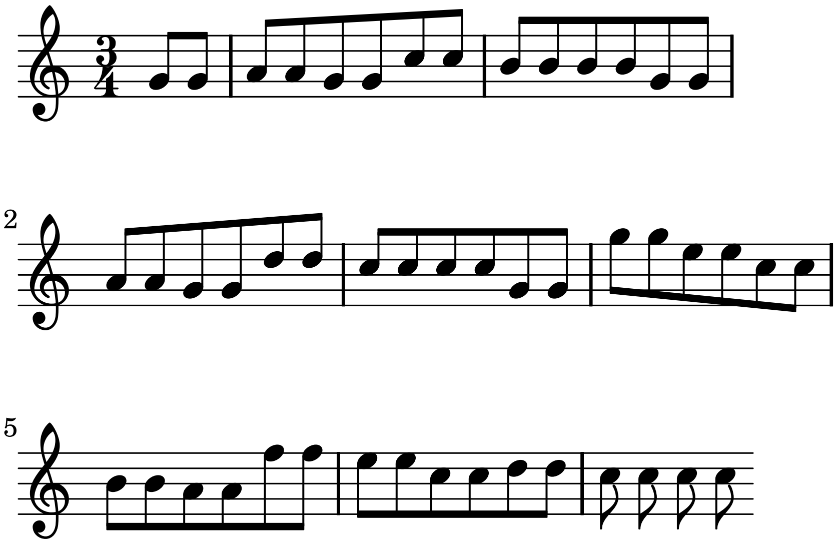

Fig. 2. The interpretation of the computation of the P system Π_{HB}

Acknowledgement

The second author acknowledges the support of the projects TIN2008-04487-E and TIN-2009-13192 of the Ministerio de Ciencia e Innovación of Spain and the support of the Project of Excellence with *Investigador de Reconocida Valía* of the Junta de Andalucía, grant P08-TIC-04200.

References

1. M. Cardona, M.A. Colomer, A. Margalida, I. Pérez-Hurtado, M.J. Pérez-Jiménez, D. Sanuy. A P system based model of an ecosystem of some scavenger birds. *Lecture Notes in Computer Science*, **5957** (2010), 182-195
2. M. Cardona, M.A. Colomer, M.J. Pérez-Jiménez, D. Sanuy, A. Margalida. Modeling ecosystem using P systems: The bearded vulture, a case study. *Lecture Notes in Computer Science*, **5391** (2009), 137-156.
3. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez. A Simulator for Confluent P Systems. Third Brainstorming Week on Membrane Computing. RGCN Report 01/2005 Fénix Editora, Sevilla (Spain), (2005), 169-184 .
4. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez, F.J. Romero-Campero. How to express tumours using membrane systems. *Progress in Natural Science*, **17**(4) (2007), 449-457.

5. R. King, C. Angus. PM - Protein Music. *Computer Applications in the Biosciences*, **12**, (1996), 251-252.
6. C. Miner F. Della Villa. DNA Music. *The Science Teacher*, **64**(5), (1997), 19–21.
7. M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini. A polynomial complexity class in P systems using membrane division. Proceedings of the 5th Workshop on Descriptional Complexity of Formal Systems, DCFS 2003, Computer and Automaton Research Institute of the Hungarian Academy of Sciences (2003), pp.:284-294.
8. M.J. Pérez-Jiménez, F.J. Romero-Campero. A study of the robustness of the EGFR signalling cascade using continuous membrane systems. *Lecture Notes in Computer Science*, **3561** (2005), 268-278.
9. A. Romero-Jiménez, M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez. Graphical Modelling of Higher Plants Using P Systems. *Lecture Notes in Computer Science* **4361**, (2006) 496-506.