Membranes as Multi-agent Systems: an Application to Dialogue Modelling

Gemma Bel-Enguix and Dolores Jiménez López

Research Group on Mathematical Linguistics
Rovira i Virgili University
Pl. Imperial Tàrraco, 1, 43005 Tarragona, Spain
{gemma.bel,mariadolores.jimenez}@urv.net

Abstract. Human-computer interfaces require models of dialogue structure that capture the variability and unpredictability within dialogue. In this paper, taking as starting point P systems, and by extending it to the concept of dialogue P systems through linguistic P systems we introduce a multi-agent formal architecture for dialogue modelling. In our model, cellular membranes become contextual agents by the adjunction of cognitive domains. With this method, the passage from the real dialogue to the P systems model can be achieved in a quite intuitive and simple way.

1 Motivation

Computational work on discourse has focused both on extended texts and on dialogues. Work on the former is relevant to document analysis and retrieval applications, whereas research on the latter is important for human-computer interfaces. In fact, the development of machines that are able to sustain a conversation with a human being has long been a challenging goal. If we focus on dialogue modelling, we can distinguish two related research goals adopted by researchers on the field: 1) to develop a *theory of dialogue* and 2) to develop *algorithms* and procedures to support a computer's participation in a cooperative dialogue.

In this paper we introduce a *formal theory* of dialogue based in a model – P systems – already introduced in theoretical computer science, which should be easily implemented. Our aim is to introduce a theoretical model of conversation with the explicitness, formality and efficiency that are required for computer implementation.

The essential characteristic of the model is the use of a simple computational mechanism for the interaction between cellular membranes in order to generate a conversation structure. So, dialogue P systems can be placed in the line of those approaches to modelling conversation that use the so-called *dialogue grammars*, a useful computational tool to express simple regularities of dialogue behaviour.

P systems –introduced in [10]– are models of computation inspired by some basic features of biological membranes. They can be viewed as a new paradigm

in the field of natural computing based on the functioning of membranes inside the cell.

Membranes provide a powerful framework for formalizing any kind of *interaction*, both among agents and among agents and the environment. An important idea in P systems is that generation is made by evolution, when the configuration of membranes undergoes some modifications, given by certain rules. Therefore, most of evolving systems can be formalized by means of membranes.

P systems have been already applied to linguistics in [2, 3] and other suggestions for different specific implementations have been given in [11].

The most important intuition for translating this natural computing model to natural languages is that membranes can be understood as *contexts* or more general environments, which may may be different words, persons, social groups, historical periods, languages. In this model agents can accept, reject, or produce changes in elements they have inside. At the same time, contexts/membranes and their rules evolve, that is, change, appear, vanish, etc. Therefore, membranes and elements of the system are constantly interacting.

In this paper, we suggest the a type of multi-agent P structures for dealing with dialogue modelling, since this is a topic where context and interaction among agents is essential to the design of effective and user-friendly computer dialogue systems.

In section 2, the definition of P systems is given, as well as an adaptation of the computational model to deal with linguistics, linguistic P Systems. Section 3 explains how, because of their flexibility, membranes can be used to model different parts of linguistics, being focalized in this paper for studying dialogue. In section 4 dialogue P Systems are introduced. In 5, we give some conclusions and lines of research for the future work.

2 P Systems and Linguistic P Systems

P systems, as a computational model based in biology, consist of multisets of objects which are placed in the compartments defined by the membrane structure –a hierarchical arrangement of membranes, all of them placed in a main membrane called the *skin membrane*– that delimits the system from its environment.

In general, P systems are a distributed parallel computational model based in the concept of *membrane structure*. Such structure is represented by a Venn diagram where all the sets, *membranes*, are inside a unique *skin membrane*. A membrane without any membrane inside is called *elementary membrane*. Every membrane delimits a *region*. *Objects*, placed in these regions, are able to evolve travelling to other membranes or being transformed in different objects.

Membranes are usually represented by the sign [ ], and they are labelled with a number between 1 and the number n of membranes in the system. For example, the structure in Figure 1 has the shape $[\,[\,]_2\ [\,]_3\ [\,[\,]_5\,[\,[\,]_8\,[\,]_9\,]_6\,[\,]_7\,]_4\,]_1$.

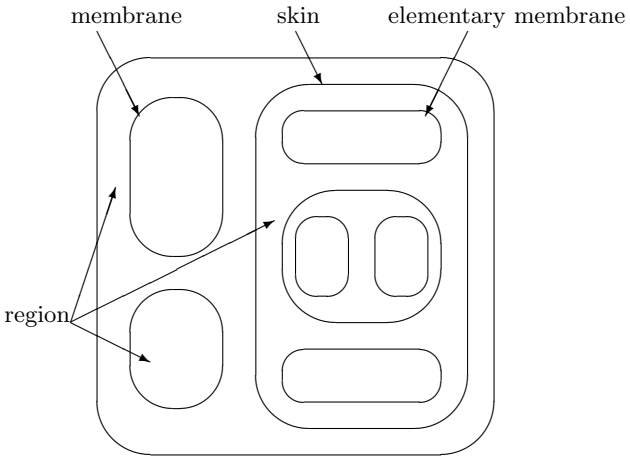

Fig. 1. A membrane structure.

Formal definitions and main issues related to the topic can be found in [10] and [11]. Nevertheless, we introduce the basic description of membranes, which can be useful to understand the relationship between general P systems and P systems for dialogue.

Definition 1 *A P system Π is defined as a construct*

$$\Pi = (V, \mu, w_1, ...w_n, (R_1, \rho_1), ..., (R_n, \rho_n), i_o),$$

where:

- *V is an alphabet; its elements are objects;*
- *μ is a membrane structure of degree n;*
- *w_i, $1 \leq i \leq n$, are strings from V^* representing multisets over V associated with the regions 1, 2,...n of μ;*
- *R_i, $1 \leq i \leq n$, are finite sets of evolution rules over V associated with the regions 1, 2,...n of μ; ρ is a partial order relation over R_i, $1 \leq i \leq n$, specifying a priority relation between rules of R_i.*
- *i_o is a number between 1 and n, which specifies the output membrane of Π.*

In membranes, rules of any type can be applied in any membrane, and the results can be sent to other membranes, increasing the computational power and efficiency of the model.

The adaptation of P systems to linguistics [2, 3] gave rise to linguistic P systems (LPS). Although the formalization is mainly the same, the aim of LPS is not to produce languages, but to model linguistic processes. The chief formal differences of LPS with regard to usual membrane systems are: a) they are not

always parallel systems, b) they use more than one alphabet, c) membranes have domains which represent different contexts, in a way that membranes can accept objects or not depending on the domain.

Domains are an important feature of linguistic P systems. The domain D of a membrane M_n is a set over V associated to the membrane in every state of the computation. It is the set of semantemes this membrane accepts. The subscript $_i$ attached to an object in a membrane M_n means that it is not accepted by M_n because it is not in its domain. From here, we infer that domains can be understood as contexts or worlds.

From here, a definition of LPS can be given.

Definition 2 *A linguistic P system Π is defined as a 4-uple*

$$\Pi = \{\mu, V, I, R\},$$

where

- μ *is the membrane system of degree* n,
- $V = \{V_1, \ldots, V_i\}$ *is the set of alphabets associated to each membrane,*
- $I = (\{u \ldots w\}, C, D, t)$ *is the initial configuration of each membrane,*
- $R = \{R_1, \ldots, R_n\}$ *is the set of rules of every membrane of the system, including*
  - *evolutionary rules for alphabets*
  - *evolutionary rules for membranes*

To fully understand the working of such systems, several aspects have to be explained: a) relations between membranes, b) communication in μ, c) operations with membranes.

Concerning the point a), the way the membranes are related to the others is important in the moment they have to interact, and also in the configuration of the communication we are going to deal with later. There are mainly tree types of relations: *nesting*, *adjacency* and *command*.

Given two membranes M_1, M_2, it is said M_2 to be *nested* in M_1 when it is inside M_1. The outer membrane M_1 is called *parent membrane* and the inner membrane M_2 is called *nested membrane*. It is denoted $M_2 \subset M_1$: $[_1\ [_2\ ]_2\ ]_1$. *Degree of nesting* refers to the number of membranes between the nested one and the parent one. The degree of nesting is obtained by subtracting the depth of the parent membrane M_p to the depth of the nested membrane M_n. This is: $deg(M_n \subset M_p) = depth(M_n) - depth(M_p)$

Two membranes M_n, M_m are related by *sibling*, if they satisfy:
i. they have a common parent membrane, and
ii. they have the same depth.

Sibling is denoted $M_n \approx M_m$. Namely, in a membrane system denoted as $[_0\ [_1\ [_2\ ]_2\ ]_1\ [_3\ [_4\ ]_4\ ]_3\ ]_0$, $M_1 \approx M_3$ and $M_2 \approx M_4$.

Given two membranes M_n, M_m, M_n *commands* M_m iff:
i. they are not nested,

ii. both are nested in a membrane M_j,
iii. $deg(M_n \subset M_j) = 1$, $deg(M_m \subset M_j) > 1$

Command is denoted $M_n \triangleleft M_m$. Considering the existence of a system with the structure $[_1\ [_2\ ]_2\ ]_1\ [_3\ [_4\ ]_4\ ]_3$, $M_1 \triangleleft M_4$ and $M_3 \triangleleft M_2$.

In what concerns the communication between membrane in the system, if membranes are understood as social groups, conditions of marginality or integration may be modelled by means of the connection/non-connection between a concrete group and the others. If membranes refer to different agents in a dialogue, then it is easy to find agents which do not participate, wher eas some others keep the attention all the time.

These concepts can be approached by means of the introduction of communication channels between membranes. We establish that, between two membranes there is always a communication channel, that can be open $\odot$ or closed $\otimes$. Every membrane has the control over its communication channels – except the skin membrane, which is always connected.

For the communication to be possible between two given membranes M_n, M_m to be open, it is necessary for that channel to be open from M_n and M_m, what can be represented by $M_n \odot \odot M_m$. The other possibilities: $M_n \odot \otimes M_m$, $M_n \otimes \odot M_m$, $M_n \otimes \otimes M_m$, do not allow the exchange of information. It can be said, then, that the communication is bidirectional, and it is only possible if it is allowed from both sides.

When a membrane has every communication channel open, then it is said to be *fully connected*. It is represented by $\odot M_n$. When a membrane has every communication channel closed, then it is said to be *inhibited*. It is denoted by $\otimes M_n$. The states of the communication channels can be set and changed by specific rules during the computation.

Graphically, connection between two membranes is not represented, while inhibition is drawn with a double line for the membrane. If two membranes are not connected because one of them has closed that channel, it is represented by two parallel lines.

Concerning operations that can be applied in membranes during the computation, structure of P systems can undergo several variations. Some contexts can disappear or be extended during the progress of the conversation, others can merge, or be copied many times. The flexibility of LPS requires the formalization of some rules regulating different ways of interaction in the main components of P Systems, membranes. These operations are the ones defined in the sequel.

1. Dissolution. By means of deletion a membrane M_n is dissolved and its elements go to the immediately external membrane. The rule for deleting membrane M_n is written as: $[\ [\ v]_m]_n \Rightarrow [\ v]_n$.

2. Deletion. It is the operation by means of which a membrane M_n completely disappears with all its elements. The rule is $[\ [\ v]_n]_m\ \Rightarrow [\ ]_m$.

3. Merging. By merging, two adjacent membranes M_n, M_m join in just one by the rule: $[\ [u]_n\ [v]_m\ ] \Rightarrow [\ [uv]_j]$

4. Splitting By means of division, a membrane M_m is divided in two or more. Graphically, we write $[\ [uw]_m] \Rightarrow [\ [uv]_m\ [uv]_n]$.
5. Extraction. It is the operation by means of which a membrane nested in another one is extracted, being both related by sibling of degree 0 in the resulting configuration. It is denoted by $[\ [\ [u]_n\ ]_m\ ] \Rightarrow [\ [\ ]_m[u]_n\ ]$.
6. Insertion. It is the inverse operation of extraction. By thgis operation a membrane wich is adjacent to another one is nested in it with degree 0. It is denoted by $[\ [\ ]_m[u]_n\ ] \Rightarrow [\ [\ [u]_n\ ]_m\ ]$.

3 Different applications of LPS to linguistics

We think that the general definition of LMS that has been given introduces a quite flexible tool for modelling several linguistic aspects, especially the ones not directly related to syntax. To be adapted in order to suitably model these linguistic branches, only some aspects must be adjusted in the general description for them to be optimal for each field. The most important is the different interpretation of membranes that is allowed by the introduction of the notion of *domain*. But also the elements of the alphabets can be understood differently depending on the interpretation of the domains. We consider also the existence of turn-taking, which is on of the key points for modelling dialogue, and the different treatment of the output in each one of the systems.

Taking into account these variables, a table can be composed, which shows several parts of linguistics that can be approached by this method.

	Domains	Elements	TT	OM
Semantics	Contexts	Linguemes	No	μ
Lang. evolution	Languages	Lgc. units	No	μ
Sociolinguistics	Social groups	Lgc. units	No	μ
Dialogue	Competence	Speech Acts	Yes	CR
Anaphora resol.	Contexts	Anaphores	No	i_o

Table 1. Features of several applications on LMS in linguistics

From the results above, we are interested in the modelling of dialogue. Therefore, domains are interpreted as the personal background and competence of the agent, which includes concepts such as education, context, and knowledge of the world. The basic elements for generating the dialogue are speech acts [1, 14, 6] and a method for assigning the turn-taking has to be introduced. Finally, the output membrane for dialogue systems does not exist, and it is substituted by a Generation Register (CR)

In the sequel we deal with an example of representation of non task-oriented (NTO) dialogues by means of membranes.

4 Dialogue P Systems

Now, before going on with the introduction of the computing device, we have to introduce the main units in the system we want to define. Basic elements we deal with in here are *speech acts*. An speech act can be defined as a communicative event whose final meaning is not only related to syntax but also to the illocutionary strength of the speaker. Speech acts has been traditionally a central topic in pragmatics, now they have an increasing importance in the so-called dialogue games [5, 8], an attempt to start a formal study of pragmatic situations.

Combining both theories, several classifications of conversation act types have been given (cf. [15]). We just want to take into account a small list of acts including the most usual ones. The goal is to have a set of utterances to test the suitability of the model, since the final objective of this paper is not to discus about the taxonomy of acts in dialogue.

Therefore, adapting some general concepts to a *computational description*, we propose to distinguish the following types of acts in human communication: 1) *Query-yn* (yes, no), 2) *Query-w* (what), 3) *Answer-y* (yes), 4) *Answer-n* (no), 5) *Answer-w* (what), 6) *Agree*, 7) *Reject*, 8) *Prescription*, 9) *Explain*, 10) *Clarify*, 11) *Exclamation*. This list may be modified any moment depending on the convenience and accuracy of the theory.

Acts are usually gathered in topics during a conversation. For starting, closing, or changing a topic, some special expressions are usually used. They are *structural acts*, and should be added to the above list of sequences, obtaining: 12) *Open*, 13) *Close*, 14) *Changetopic*. Structural acts have some special features which make them different. *Open* is the first act, or at least the first instruction, in every dialogue or human interaction. However, *close* is not always present, in the same way that, many times, topics are not closed in conversations, and new ones arise without and ending for the previous. On the other hand, *changetopic* is a sequence of transition which cannot be followed by every agent.

Nevertheless, these concepts have to be adapted to the diversity of realistic situations, which may be quite unexpected. In a dialogue, not every agent has every type of speech act. Depending on the *competence* of each agent, some speech acts can be blocked. For instance, only an agent with certain power can use the act *prescription*. The distribution of speech acts among the agents will be very important in the development of the dialogue.

Definition of dialogue P systems (in short, DPS) is based in the general formalization of membrane systems introduced in Section 2, adding a special treatment for the turn-taking and a Generation Register (GR) for storing the output generated by the system.

Definition 3 *A dialog P system is a 5-uple,*

$$\Pi = (\mu, V, I, T, R),$$

where:

- μ *is the membrane system;*
- $V = \{V_1, \ldots, V_i\}$ *is the set of alphabets associated to types of speech acts;*
- $I = (\{u \ldots w\}, C, D, t)$ *is the initial configuration of each membrane, being:*
  - $\{u \ldots w\}$ *the set of acts over* $V^\star$*;*
  - D*, the domain of the membrane;*
  - C*, the communication state of the membrane;*
  - t *is any element of* T*.*
- T *is the turn-taking set;*
- $R = \{R_1, \ldots, R_n\}$ *is the set of rules of every membrane of the system, where the order in which rules are given is also a preference for using them.*

Several concepts should be explained for clarifying the description and working of the system: a) configuration of alphabets, b) shape of the rules, c) domains, d) the turn-taking protocol T, e) halting criteria, and f) configuration of the output.

Configuration of Alphabets. Basic elements of DPS are speech acts. These speech acts are gathered in several types, following the classification given above. Every one of these types is an ordered set of elements which can be used just one time, according to the precedence.

We define a set of alphabets $V = \{\omega, \#, \kappa', \kappa, \alpha^y, \alpha^n, \alpha, \gamma, \varphi, \tau, \varepsilon, \lambda, \xi\}$, where every element is a set of speech acts, as follows: $\omega = \{o_1, o_2, ..o_n\}$, speech acts of type *open*; $\# = \{\#_1, \#_2, ..\#_n\}$, speech acts of type *close*; $\kappa' = \{q'_1, q'_2, ..q'_n\}$, speech acts of type *query-yn*; $\kappa = \{q_1, q_2, ..q_n\}$, speech acts of type *query-w*; $\alpha^y = \{a^y_1, a^y_2, ..a^y_n\}$, speech acts of type *answer-y*; $\alpha^n = \{a^n_1, a^n_2, ..a^n_n\}$, speech acts of type *answer-n*; $\alpha = \{a_1, a_2, ..a_n\}$, speech acts of type *answer-w*; $\gamma = \{g_1, g_2, ..g_n\}$, speech acts of type *agree*; $\varphi = \{f_1, f_2, ..f_n\}$, speech acts of type *reject*; $\pi = \{p_1, p_2, ..p_n\}$, speech acts of type *prescription*; $\varepsilon = \{e_1, e_2, ..e_n\}$, speech acts of type *explain*; $\lambda = \{l_1, l_2, ..l_n\}$, speech acts of type *clarify*; $\xi = \{x_1, x_2, ..x_n\}$, speech acts of type *exclamation*.

Shape of the rules Rules are understood as the way the membranes-agents exchange elements and interact each other. Every rule in the system has at the left side the indication of the turn-taking. At the right side it has, a) the generation in reply to the explicit invitation to talk (turn-taking element), and b) the agent whom the speech act is addressed to, if it exists.

The turn-taking allows applying just one rule.

Domains In DPS, *domain* of a membrane is related to the competence of an agent in a dialogue, this is, what the agent knows and can say. It is defined as the set of speech acts that every membrane is able to utter. It can include entire sets of acts defined for the system or just single acts coming from some set. Of course, just speech acts defined in V, this is, existing in the possible world described by μ, can be used. $DM_n = \{u, .., w \in V\}$.

Turn-Taking Protocol For dialogues we are dealing with, turn-taking must be free, this is, it is not given as a sequence, but as a set of active elements, at the beginning of the computation. Every turn is distributed by the agent that is talking. When somebody asks, explains, or clarifies something, in a dialogue, he/she does it to somebody among the others. Then, we establish that the addresser in each turn can choose next speaker. It does it by means of the *turn-taking rule* included at the end of each rule of the system. This is denoted by means of a letter depending of the speech act uttered in such rule.

Therefore, the following turn-taking symbols related to every speech act are considered : O (open), # (close,) Q' (Query-yn), Q (Query-w), A^y (Answer-y), A^n (Answer-n), A (Answer-w), G (Agree), F (Reject), P (Prescription), E (Explain), L (Clarify), X (Exclamation), H (Changetopic).

We include H for *changetopic* among these symbols, which is not related to any set of speech acts, because any type (except answer) can be a good reply to it. If no indication of turn is given in a rule, the turn goes to every membrane able to reply, this is, every membrane containing a rule with the required symbol in the left. If there are several membranes able to act, then the turn is indicated by the number of the membrane, which also establishes an order of precedence in the computation, this is $M_1 < M_2 < M_3 < .. < M_n$.

Halting Criteria We establish that the system stops if one of the following conditions is fulfilled: a) No rule can be applied in any membrane, b) just one membrane remains in the system, c) no more acts are available.

Configuration of the Output For DPS there are not output membranes. For the *configuration of the output*, we define the Generation Register (GR). The generation register gives account of the changes in the configuration of the system in every step. To look at the GR is the way to know what the final result of the system is.

5 Final Remarks

In this paper a new approach for a formal modelling of human communication in the framework of pragmatics using P systems has been introduced and several basic issues related to a membranes approach to conversation have been developed in order to test the suitability of the model.

We think that to apply P systems to linguistic topics has several advantages, among which we stress the flexibility of the model. Considering membranes as agents, and domains as a personal background and linguistic competence, the application to dialogue is almost natural, and simple from the formal point of view. Many variations can be introduced to the basic model presented in this paper in order to account for different features of conversation, and this can be a good research area for the future.

Nevertheless, since this is just an initial approximation to the possibility of describing conversation by means of membrane systems, many important aspects remain to be approached: formalization of task-oriented and institutional conversations, non-free turn-taking, interactions among ag ents, introduction of different conversation act types or modelling of parallel phenomena.

Finally, although the model is defined for formally describing human communication, we think that it can be applied to the generation of conversations in the framework of human-computer or computer-computer interface.

References

1. Austin, J.L. (1962), *How to Do Things With Words*, New York, Oxford University Press.
2. Bel Enguix, G. (2003), Preliminaries about Some Possible Applications of P Systems in Linguistics, in Păun, Gh., Rozenberg, G., Salomaa, A. & Zandron, C., *Membrane Computing*, Springer, Berlin.
3. Bel Enguix, G. & Jiménez López, M.D. (2005), Linguistic Membrane Systems and Applications, in Ciobanu, G., Păun, Gh. & Pérez Jiménez, M.J. (eds.), *Applications of Membrane Computing*, Springer, Berlin, pp. 347-388.
4. Bunt, Harry C. (1990), *DIT-Dynamic Interpretation in Text and Dialogue*, ITK Research Report, no. 15, Tilburg University, The Netherlands.
5. Carlson, L. (1983), *Dialogue games*, Dordretch, Reidel.
6. Grice, H.P. (1975), Logic and Conversation, in Cole, P. & Morgan, J. (eds.) *Syntax and Semantics 3: Speech Acts*, New York: Academic Press.
7. Jiménez-López, M.D. (2002), Formal Languages for Conversation Analysis, in García Español, A. (ed.), *Estudios Hispánicos y Románicos*, Universitat Rovira i Virgili, Tarragona.
8. Kwotko, J.C., Isard, S.D. & Doherty G.M. (1993), Conversational Games Within Dialogue, *Technical Report HCRC/RP-31*, HCRC Publications University of Edinburgh.
9. Morris, C. (1938), Foundations of the Theory of Signs, in Carnap, R. et al (eds.), *International Encyclopaedia of Unified Science*, 2:1, Chicago, The University of Chicago Press.
10. Păun, Gh. (2000), Computing with Membranes, *Journal of Computer and System Sciences*, 61, pp. 108-143.
11. Păun, Gh. (2002), *Membrane Computing. An Introduction*, Springer, Berlin.
12. Reed C.A., Long D.P. (1997), Collaboration, Cooperation and Dialogue Classification, in Jokinen, K. (ed) *Working Notes of the IJCAI97 Workshop on Collaboration, Cooperation and Conflict in Dialogue Systems*, Nagoya, pp. 73-78.
13. Sacks, H., Schegloff, E.A., & Jefferson, G. (1974), A Simplest Systematics for the Organization of Turn-Taking for Conversation, *Language*, 50, 4, pp. 696-735.
14. Searle, J. (1969), *Speech Acts: An Essay in the Philosophy of Language*, Cambridge University Press.
15. Traum, D.R. *Speech Acts for Dialogue Agents*, in Wooldride, M. & A. Rao (eds.), "Foundations of Rational Agency", Kluwer, Dordrecht, 1999, 169-201.