Some Notes on (Mem)Brane Computation

Mura Anteo Zamboni 7, I-40127 Bologna, Italy
busi@cs.unibo.it

[2] Dpto. de Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla
Avda. Reina Mercedes s/n, 41012, Sevilla, Spain
magutier@us.es

Abstract. Membrane Computing and Brane Calculi are two recent computational paradigms in the framework of Natural Computing. They are based on the study of the structure and functioning of living cells as living organisms able to process and generate information. In this paper we give a short introduction to both areas and point out some open research lines.

1 Introduction

Natural Computing studies new computational paradigms inspired from various well known natural phenomena in physics, chemistry and biology. This paper is devoted to a new field in Natural Computing: The study of the structure and functioning of cells as living organisms able to process and generate information.

The starting point is the fact that the cell is the smallest living thing, and at the same time it is a marvellous machinery, with a complex structure, an intricate inner activity self-regulated in a quite efficient way. Assuming that cells can be seen as computational devices, two different branches of Natural Computing can be found in the literature: *Membrane Computing* and *Brane Calculi*.

The notions of membrane investigated in these new paradigms of computation are abstract entities which try to mimic some features of the functioning of membranes in living cells. The basic function of biological membranes is to *define compartments* and to *relate compartments to their environment*, including neighbouring compartments. The currently accepted model of the membrane structure is the so-called *fluid-mosaic model*, proposed in 1972 by S. Singer and G. Nicholson. According to this model, a membrane is a phospholipid bilayer in which protein molecules (as well as other molecules) are totally or partially embedded.

The first paradigm of computation we present is *Membrane Computing*. It was introduced by Gh. Păun in [22] under the assumption that the processes

⋆ The second author acknowledges the support by project TIN2005-09345-C04-01 of Ministerio de Educación y Ciencia of Spain, cofinanced by FEDER funds, and by the Project of Excellence TIC-581 of the Junta de Andalucía.

taking place in the compartmental structure of a living cell can be interpreted as computations. The devices of this model are called *P systems*. Roughly speaking, a P system consists of a membrane structure, in the compartments of which one places multisets of objects which evolve according to given rules in a synchronous nondeterministic maximally parallel manner.

The second one, *Brane Calculi* was introduced by L. Cardelli in [9] on the assumption that in living cells membranes are not merely containers, but they are highly dynamic entities that actively participate in the cell life. In this way, computation happens on the membrane, not inside it.

The paper is organised as follows: Section 2 is devoted to a brief presentation of Membrane Computing. In a similar way, a short introduction to Brane Calculi is presented in Section 3. The paper ends with some final remarks.

2 Membrane Computing

Membrane Computing[1] starts with the explicit goal of abstracting computing models from the *structure* and the *functioning* of a living cell. The literature of the domain is very large (already in 2003, Thompson Institute for Scientific Information, ISI, has qualified the initial paper as "fast breaking" and the domain as "emergent research front in computer science" – see `http://esi-topics.com`) and it progresses rather rapidly, so that the presentation here is quite general.

The basic idea is to consider a distributed and parallel computing device, structured like a cell, by means of a hierarchical arrangement of membranes which delimit compartments where various chemicals (we call them *objects*) evolve according to local reaction rules. The objects can be described by symbols or by strings of symbols from a given alphabet. These objects can also pass through membranes, under the control of specific rules. Because the chemicals from the compartments of a cell are swimming in an aqueous solution, the data structure we consider is that of a *multiset* – a set with multiplicities associated with its elements. Also, in close analogy with what happens in a cell, the reaction rules are applied in a parallel manner. This means that in each computational step a maximal (multi)set of nondeterministically chosen rules is applied.

The *membrane structure* of a P system is a hierarchical arrangement of membranes (understood as three dimensional vesicles), embedded in a *skin* membrane, the one which separates the system from its *environment*. A membrane without any membrane inside is called *elementary*. Each membrane defines a *region*. For an elementary membrane this is the space enclosed by it, while the region of a non-elementary membrane is the space in-between the membrane and the membranes directly included in it. Membranes are labelled. Each region contains a multiset of *objects*, and a set of (*evolution*) *rules*. The objects are represented by symbols from a given alphabet. Typically, an evolution rule from region r is of the form $ca \to cb_{in_j} d_{out} d_{here}$, and it "says" that a copy of the

[1] A layman-oriented introduction can be found in [30], a comprehensive presentation can be found in [23] and further updated bibliography in [34]. A presentation of applications can be found in [14].

object a, in the presence of a copy of the *catalyst* c (this is an object which is never modified, it only assists the evolution of other objects), is replaced by a copy of the object b and two copies of the object d. Moreover, the copy of b has to enter "immediately" the inner membrane of region r labelled by j (hence to enter region j), one copy of object d is sent out through the membrane of region r, and one copy of d remains in region r. Note that the considered evolution rule can be applied in the region r only if this region includes the membrane j.

Membrane systems are *synchronous*, in the sense that a global clock is assumed. In each time unit a transformation of a *configuration* of the system takes place by applying the rules in each region, in a *nondeterministic* and *maximally parallel manner*. This means that the objects to evolve and the rules governing this evolution are chosen in a nondeterministic way; this choice is "exhaustive" in the sense that, after the choice was made, no rule can be applied anymore in the same evolution step (there are not enough objects available anymore for any rule to be applied now – this is the maximality of application). In this way, one gets *transitions* between the configurations of the system. A maximal sequence of transitions is called a *computation*. A configuration is *halting* if no rule is applicable in any region. A computation is *halting* if it reaches a halting configuration. The *result* of a (halting) computation is the *number* of objects sent (through the skin membrane) to the environment during the computation.

Many variants/extensions of this very basic model sketched above are discussed in the literature.

2.1 Variants of the Basic Model

Here we will briefly mention a few variants of the basic model. For instance, there are a number of ways of weakening the programming power provided by in_j: to only indicate *in* (an object associated with this command has to enter any adjacently lower membrane; the choice of a membrane to enter is nondeterministic), to associate *electrical charges* both with objects and with membranes (a polarised object will enter the region of any adjacently lower membrane of the opposite polarisation; the polarisation of objects and of membranes may change during the computation).

Coming closer to the trans-membrane transfer of molecules, we can consider purely communicative systems, based on the three classes of such transfer known in the biology of membranes: *uniport, symport*, and *antiport*. Symport refers to the transport where two molecules pass together through a membrane in the same direction, antiport refers to the transport where two molecules pass through a membrane simultaneously, but in opposite directions, while the case when a molecule does not need a "partner" for a passage is referred to as uniport.

Another important extension is to consider a priority relation among rules. Furthermore, we can have rules for handling membranes (creating, destroying, dividing, merging, etc.), the rules can have promoters or inhibitors, their use can be regulated by a priority relation, the permeability of membranes can be controlled by the used rules and so on and so forth, either with a biological or with a mathematical motivation. In short, we abstract as much as possible/necessary,

in order to obtain a mathematical model which is intended to be (i) minimalistic (as elegant as possible, containing as restricted ingredients as possible), but (ii) without losing the biological inspiration (hence remaining as "realistic" as possible), with (iii) good computability properties (as *powerful* as possible and as *efficient* as possible).

2.2 Computational Power and Efficiency

Many classes of P systems, combining various ingredients described above, are capable of simulating Turing machines, hence they are *computationally complete*. Note that in the case when we deal with P systems which compute numbers, we consider Turing machines as number recognisers; in the case of string-objects we can obtain the family of languages which are recognised by Turing machines (the recursively enumerable languages). Always, the proofs of results of this type are constructive, and this has the important consequence from the computability point of view that we can get *universal* (hence *programmable*) P systems: starting from a universal Turing machine, we get an equivalent universal P system.

The computational power is one of the important questions to be dealt with when defining a new computing model. The other fundamental question concerns the computational *efficiency*.

One of the explicit goals of various branches of natural computing is to find ways to address computationally hard problems (typically, **NP**-complete problems) in order to solve them (in a strict sense or in a probabilistic sense) in a feasible (that is, in polynomial) time. The rules of a P system are used in parallel, that is, in each membrane all objects evolve simultaneously, and, in turn, at the level of the system all membranes evolve simultaneously. This is a good degree of parallelism, which, however, is not sufficient to devise polynomial time solutions to **NP**-complete problems (unless $\mathbf{P} = \mathbf{NP}$). However, biology suggests operations with membranes which, sometimes surprisingly, make possible polynomial (often linear) solutions to **NP**-complete problems. Among these operations, the most investigated so far in membrane computing have been membrane division and membrane creation.

We do not enter here into details, but we refer, e.g., to the chapter from [14] devoted to this topic. Anyway, these results are of a clear theoretical interest (new characterisations of the standard complexity classes were given, as well as a characterisation of the relation $\mathbf{P} = \mathbf{NP}$ problem, intriguing borderlines between efficiency and non-efficiency were found – with many challenging open problems still waiting to be considered).

2.3 Applications

As mentioned above, Membrane Computing was initiated having as primary goals computability in general, and Natural Computing in particular, without aiming to faithfully model biological facts in such a way as to provide a modelling framework for the use of biologists. However, after significantly developing at the theoretical level, the domain started to be useful for biological and medical applications.

For example, the modelling of some dynamical systems, where we are not interested in halting configurations, but in the evolution of the process itself (see [34] and the corresponding chapter from [14]).

Another important application field is the study of processes related to cancer. We only mention the simulation of p53 protein pathways control (the interaction between proteins p53 and MdM2) through a P system, as carried out by Y. Suzuki and his co-workers (details can be found in [14]), and the modelling of EGFR (epidermal growth factor receptor) signalling network [31].

A very promising application is the study of approximate algorithms for hard optimisation problems (see [20]). These algorithms can be considered as high level (distributed and dynamically evolving their structure during the computation) evolutionary algorithms. The strategy has been checked for the travelling salesman problem and the results were more than encouraging.

Besides applications in biology, membrane computing has also been considered in other areas, such as computer graphics [33], cryptography [19], modelling in a uniform way parallel architectures [12], economics [25,26], etc. Some theoretical applications of (notions and ideas central to) P systems has also been considered in several papers: to artificial life [18], for simulating the photosynthesis [21], to linguistics [2], etc.

3 Brane Calculi

In recent years, the modeling and analysis of the biological matter has attracted the interest of the researchers in the area of concurrent process calculi.

Indeed, a network of biochemical cells can be seen as a computing machinery, made of processing agents which interact and cooperate to achieve a common goal. This informal description applies to concurrent system as well, hence it is natural to use techniques from the concurrency theory field to study the behaviour of biological cells.

Particularly promising is the use of process calculi, which are formalisms used to describe concurrent and mobile systems. Process calculi are equipped with a formal semantics describing their behaviour, and plenty of tools for the static and dynamic analysis of systems have been produced. These tools can therefore be used in the field of biological organisms, as well.

Starting from the seminal work of Buss and Fontana [17] on the use of a process calculus for the modeling of biological entities, the field has been fruitfully explored by other research groups, either by using existing calculi or by defining new, biologically inspired calculi (see, e.g., [29,28,13,32,15,27], just to mention a few).

Brane Calculi [9] are a family of process calculi proposed for modeling the behavior of biological membranes. In a process algebraic setting, Brane Calculi represent an evolution of BioAmbients [32], a variant of Mobile Ambients [10] based on a set of biologically inspired primitives of interaction. The main novelty of Brane calculi consists in the fact that the active entities reside on membranes, and not inside membranes.

While Membrane Computing is now a well-established research field, Brane Calculi can be considered to be a newborn, rather unexplored research field. In the following, we present a brief overview of the calculi, as well as of the main existing (and ongoing) works.

In [9] two basic instances of Brane Calculi are defined: the Phago/Exo/Pino (PEP) and the Mate/Bud/Drip (MBD) calculi.

The interaction primitives of PEP are inspired by *endocytosis* (the process of incorporating external material into a cell by engulfing it with the cell membrane) and *exocytosis* (the reverse process). A relevant feature of such primitives is *bitonality*, a property ensuring that there will never be a mixing of what is inside a membrane with what is outside, although external entities can be brought inside if safely wrapped by another membrane.

As endocytosis can engulf an arbitrary number of membranes, it turns out to be a rather uncontrollable process. Hence, it is replaced by two simpler operations: *phagocytosis*, that is engulfing of just one external membrane, and *pinocytosis*, that is engulfing zero external membranes.

The primitives of MBD are inspired by membrane fusion (mate) and fission (mito).

Because membrane fission is an uncontrollable process that can split a membrane at an arbitrary place, it is replaced by two simpler operations: *budding*, that is splitting off one internal membrane, and *dripping*, that consists in splitting off zero internal membranes.

An encoding of the MBD primitives in PEP is provided in [9].

In [6] we provided a stronger separation result between PEP and MBD.

On the one hand, we showed that PEP is a Turing powerful language, by providing a deterministic encoding of Random Access Machines (a Turing–equivalent formalism).

On the other hand, we proved that the existence of a divergent (i.e., infinite) computation is a decidable property in MBD. This means that there exist no divergence-preserving encoding of PEP in MBD.

After the introduction of the two basic brane calculi PEP and MBD, containing only membranes and membrane interaction primitives, in [9] the calculus is extended with small molecules, freely floating either in the external environment or inside a membrane, and with a molecule–membrane interaction primitive.

Biological membranes contain catalysts that can cause molecules, floating respectively inside and outside the membrane, to interact with each other without crossing the membrane. Membranes can bind molecules on either sides of their surface, and can release molecules on either sides of their surface. Usually, such an operation occurs in an atomic (all-or-nothing) way. The *bind&release* operation permits to simultaneously bind and release multiple molecules.

In [4] we extend the decidability result presented in [6] in two directions. On the one hand, we showed the decidability of divergence to the calculus with molecules, and with all the molecule–membrane and membrane–membrane interaction primitives, except the *phago* operation. On the other hand, we extended the decidability result on the full calculus without *phago* to other biologically

relevant properties, such as, e.g., control state maintainability, inevitability and boundedness.

Control state maintainability can be used to check safety properties, such as, e.g., the fact that all the derivatives of a system contain at least one occurrence of a given molecule (or at least two occurrences of molecules belonging to some specified set). Inevitability can be used to check, e.g., if in all the computation a state is eventually reached that does contain no occurrences of a given molecule. Boundedness can be used to check if the number of membranes or of molecules can arbitrarily grow during the computation.

The decidability results in [4] are all constructive, i.e., they provide a computable procedure for deciding the systems properties. We plan to develop a tool for the animation and the analysis of Brane Calculus systems, also based on the results presented in this work.

We also recently started to use Brane Calculi for the modeling of biological pathways, and to apply the analysis techniques developed in [4]. A preliminary step in this direction is represented by [8], where the LDL Cholesterol Degradation Pathway [1] is modeled both in Brane Calculi and in Membrane Computing. Moreover, we also discuss an application of the analysis techniques developed in [4] to check behavioural properties of this pathway.

4 Final Remarks

In the last years, two branches of Natural Computing, *Membrane Computing* and *Brane Calculi*, have been developed, with a continuous afflux of new ideas, notions, problems, and with a series of applications, especially in modelling biological phenomena. No lab implementation was intended, and no such implementation is known to be planned for the near future[2].

Brane Calculi are somewhat dual to Membrane Computing, as they work with objects placed on membranes (corresponding to proteins attached to or embedded in the real membranes), with membranes operations controlled by these objects, and trying to stay as close to the biology as possible; also the tools and the goals are different – process algebra and systems biology, respectively.

A notable difference between Brane Calculi and P systems is concerned with the semantics of the two formalisms: whereas Brane Calculi are usually equipped with an interleaving, sequential semantics (each computational step consists of the execution of a single instruction), the usual semantics in membrane computing is based on maximal parallelism (a computational step is composed of a maximal set of independent interactions).

The fist attempt of bridging both research areas was made in [11] by the *fathers* of the disciplines L. Cardelli and Gh. Păun and as they point out Membrane Computing and Brane Calculi *have different objectives and develop in different directions. While Membrane Computing tries to abstract computing models, in*

[2] For Membrane Computing, several *simulators* have been implemented. We refer to [34] and to the corresponding chapter from [14] for details.

the Turing sense, from the structure and the functioning of the cell (...), Brane Calculi pay more attention to the fidelity to the biological reality (...).

In [11] a variant of P systems with the mate and drip operations – inspired by the corresponding primitives in Brane Calculi – is defined and proved to be Turing powerful. The *Projective Brane Calculus* [16] is a refinement of Brane Calculi, where the interaction primitives reside either on the external side or on the internal side of the membrane. In [3] a projective variant of the P systems, defined in [11], is defined and shown to be computationally complete.

In [5] the computational power of the MBD Brane Calculus equipped with two different semantics is investigated. The first semantics is the classical interleaving semantics of process calculi, whereas the second semantics is the maximal parallelism semantics used for Membrane Computing. An expressiveness gap has been found, thus confirming the intuition that the maximal parallelism semantics turns out to be a very powerful synchronization mechanism.

Recently, a bridge has been crossed the other way [7]. Instead of expressing Brane Calculi operations in terms of the Membrane Computing formalism, a problem is taken from Computer Science (the generation of the set $\{n^2 \mid n \geq 1\}$) and it is shown how it can be implemented both in Membrane Computing and in Brane Calculi.

In the last years, Membrane Computing has turned out to be a useful framework for building models with biological relevance, and the number of applications of this type is steadily increasing and becoming more and more advanced and elaborate.

This leads to considerations concerning the significance of membrane based calculi and systems (for biology, for mathematics, and for computing). The approach is clearly motivated from a mathematical point of view, not only because it is natural to (try to) model the cell computational behaviour, but also because the new computing model has a number of intrinsically interesting features. Examples of such features are: the use of multisets, the inherent parallelism, the possibility of devising computations which can solve exponential (intractable) problems in polynomial time (by making use of an exponential space created in a natural manner). At this moment, all these features are only *potentially* useful from a practical computational point of view. How should the implementation problem be approached? All these questions (and some more presented at [24]) should be explored in the future.

References

1. B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter. *Molecular Biology of The Cell.* Garland Science, 4th edition, (2002).
2. G. Bel Enguix and M.D. Jiménez-López. Linguistic Membrane Systems and Applications. In [14], 347–388.
3. D. Besozzi, N. Busi, G. Franco, R. Freund and Gh. Păun. Two Universality Results for (Mem)Brane Systems. In *Proceedings of the Fourth Brainstorming Week on Membrane Computing, Vol. I* (M.A. Gutiérrez Naranjo, Gh. Păun, A. Riscos-Núñez, F.J. Romero-Campero, eds.), Fénix Editora, (2006).

4. N. Busi. Deciding behavioural properties in Brane Calculi. *Computational Methods in System Biology 2006* (CMSB 2006), to appear.
5. N. Busi. On the Computational Power of the Mate/Bud/Drip Brane Calculus: Interleaving vs. Maximal Parallelism. Workshop on Membrane Computing 2005, LNCS 3850, (2006), 144–158.
6. N. Busi and R. Gorrieri. On the Computational Power of Brane Calculi. *Transactions on Computational Systems Biology*, LNCS, Springer, to appear. An extended abstract appeared in Proc. Computational Methods in System Biology 2005 (CMSB 2005).
7. N. Busi and M. A. Gutiérrez Naranjo. A Case Study in (Mem)Brane Computation: Generating $\{n^2 \mid n \geq 1\}$. In *Proceedings of the Fourth Brainstorming Week on Membrane Computing, Vol. I* (M.A. Gutiérrez Naranjo, Gh. Păun, A. Riscos-Núñez, F.J. Romero-Campero, eds.), Fénix Editora, (2006), 81–97.
8. N. Busi and C. Zandron. Modeling and Analysis of Biological Processes by Mem(Brane) Calculi and Systems. *Winter Simulation Conference 2006*, to appear.
9. L. Cardelli. Brane Calculi. In *Computational Methods in Systems Biology 2004* (V. Danos, V. Schachter, eds.), LNBI 3082, (2005) 257–278.
10. L.Cardelli and A.D. Gordon. Mobile Ambients. *Theoretical Computer Science*, 240(1), (2000), 177–213.
11. L. Cardelli and Gh.Păun. An Universality Result for a (Mem)Brane Calculus Based on Mate/Drip Operations. In *Cellular Computing (Complexity Aspects)* (M.A. Gutiérrez-Naranjo, Gh. Păun and M.J. Pérez-Jiménez, eds.) Fénix Editora, Sevilla, Spain, (2005). 75–94.
12. R. Ceterchi and M.J. Pérez-Jiménez. On simulating a class of parallel architectures. *International Journal of Foundations of Computer Science*, 17, 1 (2006), 91–110.
13. D. Chiargugi, M. Curti, P. Degano and R. Marangoni. VICE: A VIrtual CEll.In *Computational Methods in Systems Biology 2004* (V. Danos, V. Schachter, eds.), LNBI 3082, (2005), 207–220.
14. G. Ciobanu, Gh. Păun, M.J. Pérez–Jiménez, eds.: *Applications of Membrane Computing*. Springer, (2005).
15. V. Danos and C. Laneve. Core formal Molecular Biology. In *Programming Languages and Systems*, (P. Degano, ed.), LNCS 2618, (2003), 302–318.
16. V. Danos and S. Pradalier. Projective Brane Calculus. *Computational Methods in Systems Biology 2004* (V. Danos, V. Schachter, eds.), LNBI 3082, (2005), 134–148.
17. W.Fontana and L.W. Buss. The barrier of objects: from dynamical system to bounded organizations, Addison - Wesley, (1996), 56–116.
18. Suzuki, Y., Fujiwara, Y., Takabayashi, J. and Tanaka, H.: Artificial life applications of a class of P systems: Abstract rewriting systems on multisets. *Multiset Processing. Mathematical, Computer Science, and Molecular Computing Points of View* (C. Calude, Gh. Păun, G. Rozenberg, and A. Salomaa, eds.), LNCS 2235, (2001), 299–346.
19. S.N. Krishna and R. Rama: Breaking DES Using P Systems. *Theoretical Computer Sci.*, 299, 1-3 (2003), 495–508.
20. T.Y. Nishida. An Approximate Algorithm for NP-Complete Optimization Problems Expoiting P Systems. *Workshop on Uncertainty in Membrane Computing*, Palma de Mallorca, (2004), 185–192.
21. T.Y. Nishida. A Membrane Computing Model for Photosynthesis. In [14], 181–202.
22. Gh. Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143.
23. Gh. Păun. Membrane Computing – An Introduction. Springer-Verlag, Berlin, (2002).

24. Gh. Păun. 2006 Research Topics in Membrane Computing. In *Proceedings of the Fourth Brainstorming Week on Membrane Computing, Vol. II* (C. Graciani Díaz, Gh. Păun, A. Romero-Jiménez, F. Sancho-Caparrini, eds.), Fénix Editora, (2006), 235–251.
25. Gh. Paun and R. Paun. Membrane Computing and Economics: Numerical P Systems. *Fundamenta Informaticae*, 73, 1-2, (2006), 213–227.
26. Gh. Paun and R. Paun. Membrane computing as a framework for modeling economic processes. Submitted, (2005).
27. C. Priami and P. Quaglia. Beta binders for biological interactions. *Computational Methods in Systems Biology 2004* (V. Danos, V. Schachter, eds.), LNBI 3082, Springer-Verlag, Berlin, (2005), 20–33.
28. C.Priami, A.Regev, W.Silverman and E.Shapiro. Application of a stochastic passing-name calculus to representation and simulation of molecular processes. Information Processing Letters, 80, (2001), 25-31.
29. A. Regev and E. Shapiro. Cells as computations. Nature, 419, 343, (2002).
30. Gh. Păun and M.J. Pérez-Jiménez: Recent computing models inspired from biology: DNA and membrane computing. *Theoria*, 18 (2003), 72–84.
31. M.J. Pérez–Jiménez and F.J. Romero–Campero. A Study of the Robustness of the EGFR Signalling Cascade Using Continuous Membrane Systems. In *Mechanisms, Symbols, and Models Underlying Cognition. First International Work-Conference on the Interplay between Natural and Artificial Computation*, IWINAC 2005 (J. Mira, J.R. Alvarez, eds.), LNCS 3561, Springer, Berlin, (2005), 268–278.
32. A. Regev, E. M. Panina, W. Silverman, L. Cardelli and E. Shapiro. BioAmbients: An Abstraction for Biological Compartments. *Theoretical Computer Science*, 325, 1, (2004), 141–167.
33. A. Romero-Jiménez, M.A. Gutiérrez-Naranjo and M.J. Pérez-Jiménez. Graphical Modelling of Higher Plants Using P Systems. Accepted paper at Seventh Workshop on Membrane Computing, Leiden (The Netherlands), July 2006.
34. The P Systems Website: `http://psystems. disco.unimib.it`.