Solving a Special Case of the P Conjecture Using Dependency Graphs with Dissolution[⋆]

Alberto Leporati, Luca Manzoni, Giancarlo Mauri,
Antonio E. Porreca, and Claudio Zandron

Dipartimento di Informatica, Sistemistica e Comunicazione
Università degli Studi di Milano-Bicocca
Viale Sarca 336/14, 20126 Milano, Italy
{leporati, luca.manzoni, mauri, porreca, zandron}@disco.unimib.it

Abstract. We solve affirmatively a new special case of the *P conjecture* by Gh. Păun, which states that P systems with active membranes without charges and without non-elementary membrane division cannot solve **NP**-complete problems in polynomial time. The variant we consider is *monodirectional*, i.e., without send-in communication rules, *shallow*, i.e., with membrane structures consisting of only one level besides the external membrane, and *deterministic*, rather than more generally confluent. We describe a polynomial-time Turing machine simulation of this variant of P systems, exploiting a generalised version of dependency graphs for P systems which, unlike the original version introduced by Cordón-Franco et al., also takes membrane dissolution into account.

1 Introduction

The original variant of P systems with active membranes, which includes membrane charges (or polarisations), solves in polynomial time exactly the problems in the complexity class **PSPACE** [13]. However, the variant without charges appears to be significantly weaker. This led Păun to formulate the P conjecture in 2005 [11, Problem F], one of the long-standing open problems in membrane computing:

> *Can the polarizations be completely avoided?* [...] *The feeling is that this is not possible — and such a result would be rather sound: passing from no polarization (which, in fact, means one polarization) to two polarizations amounts to passing from nonefficiency to efficiency.*

While this general formulation of P conjecture is actually false, as P systems without charges still characterise **PSPACE** when both non-elementary membrane division and dissolution rules are allowed [2], the statement is true when dissolution rules are forbidden [6].

⋆ This work was partially supported by Fondo d'Ateneo 2016 of Università degli Studi di Milano-Bicocca, project 2016-ATE-0492 "Sistemi a membrane: classi di complessità spaziale e temporale".

The intermediate case, where dissolution is allowed but non-elementary division is not, is still open. The best known upper bound is $\mathbf{P}^{\#\mathbf{P}}$, the class of problems solved in polynomial time by Turing machines with an oracle for a counting problem [7]. However, some restricted special cases actually *do* have a **P** upper bound; this can be proved for P systems having only *symmetric* division rules [9], i.e., of the form $[a]_h \to [b]_h\ [b]_h$, or when the initial membrane structure is linear, and *only* dissolution and elementary division are allowed [14]. We refer the reader to Gazdag and Kolonits [4] for a more detailed survey of related results.

In this paper we consider another special case of the P conjecture, proving a **P** upper bound for P systems with active membranes without charges with the following three restrictions:

– *monodirectional*, that is, without send-in communication rules, as previously investigated for membranes with charges [8];
– *shallow*, that is, having only one level of elementary membranes in addition to the outermost one;
– *deterministic*, that is, having only one computation, instead of having multiple computations with the same result as in the usual *confluent* mode.

We believe that these constraints are quite natural and interesting, since monodirectional shallow deterministic P systems with active membranes have long been known to solve **NP**-complete problems in polynomial time with only two membrane charges [1]. Furthermore, they have been recently [8] proved to characterise $\mathbf{P}^{\mathbf{NP}}_{\|}$, which allows parallel queries to an **NP** oracle, with three charges[1]; clearly, the $\mathbf{P}^{\mathbf{NP}}_{\|}$ upper bound also applies to the variant without charges considered in this paper.

Among the three assumptions above, the most fundamental one is monodirectionality. As we will prove later, this restriction implies that the result of the computation only depends on at most one elementary membrane, although establishing *which one* is nontrivial. We hope, instead, to relax the latter two constraints in future works.

Besides the **P** upper bound itself, the main contribution of this paper is the tool we exploit in order to prove the upper bound. This is a generalisation of the *dependency graphs*, introduced by Cordón-Franco et al. [3] to establish the result of P systems without charges and without dissolution rules, to P systems which do include dissolving membranes. This generalisation has the potential to be extended to other variants of P systems for proving upper bounds to the complexity classes they characterise.

2 Basic Notions

The P systems analysed in this paper are *monodirectional* P systems with active membranes [8] without charges [6], using object evolution rules $[a \to w]_h$, send-

[1] The determinism of the P systems is not explicitly stated in the original paper [8], but can be easily checked by inspection of the rules.

out communication rules $[a]_h \to [\,]_h\ b$, membrane dissolution rules $[a]_h \to b$ and elementary membrane division rules $[a]_h \to [b]_h\ [c]_h$. These P systems do not use send-in communication rules $a\ [\,]_h \to [b]_h$, or division rules for non-elementary membranes, either of the “weak” form $[a]_h \to [b]_h\ [c]_h$ or the “strong” form $\big[[\,]_k\ [\,]_\ell\big]_h \to \big[[\,]_k\big]_h\ \big[[\,]_\ell\big]_h$.

Furthermore, we focus on *shallow* P systems, which have membrane structures of depth at most 1, that is, at most one level of elementary membranes besides the outermost membrane.

Finally, we require our P systems to be *deterministic*, that is, each configuration reachable from the initial one has at most one successor configuration. Notice that this condition, although much stronger than the usual confluence requirement (where multiple computations can exist, as long as they all agree on the result), does not require a single multiset of rules to be applicable at each computation step, but only that all applicable multisets of rules produce the same result.

For brevity, in this paper we refer to monodirectional, shallow, deterministic P systems as MSD P systems.

In particular, we are dealing with *recogniser P systems* [12], whose alphabet includes the distinguished result objects yes and no; exactly one result object must be sent out from the outermost membrane to signal acceptance or rejection, and only at the last computation step.

A decision problem, or language $L \subseteq \Sigma^\star$, is solved by a *family* of P systems $\boldsymbol{\Pi} = \{\Pi_x : x \in \Sigma^\star\}$, where Π_x accepts if and only if $x \in L$. In that case, we write $L(\boldsymbol{\Pi}) = L$. As usual, we require a uniformity condition [10] on families of P systems:

Definition 1. *A family of P systems $\boldsymbol{\Pi} = \{\Pi_x : x \in \Sigma^\star\}$ is* (polynomial-time) uniform *if the mapping $x \mapsto \Pi_x$ can be computed by two polynomial-time deterministic Turing machines E and F as follows:*

- *$F(1^n) = \Pi(n)$ is a common P system for all inputs of length n, with a distinguished input membrane.*
- *$E(x) = w_x$ is an input multiset for $\Pi(|x|)$, encoding the specific input x.*
- *Finally, Π_x is simply $\Pi(|x|)$ with w_x added to its input membrane.*

The family $\boldsymbol{\Pi}$ is said to be (polynomial-time) semi-uniform if there exists a single deterministic polynomial-time Turing machine H such that $H(x) = \Pi_x$ for each $x \in \Sigma^\star$.

We denote the class of problems solved by uniform (resp., semi-uniform) families of deterministic P systems of type $\mathcal{D}$ as $\mathbf{DMC}_{\mathcal{D}}$ (resp., $\mathbf{DMC}^\star_{\mathcal{D}}$). We denote the corresponding class of problems solved *in polynomial time* by $\mathbf{DPMC}_{\mathcal{D}}$ (resp., $\mathbf{DPMC}^\star_{\mathcal{D}}$).

3 Properties of MSD P Systems

We begin our analysis of MSD P systems by proving that their overall behaviour, acceptance or rejection, is actually governed by *just one or two objects*, which must moreover be located inside a single membrane. In order to be able to succinctly formalise this result, we first define a notion of restricted configuration similar, but more general than the one previously used by the authors [8, Definition 3].

Definition 2. *Given two configurations $\mathcal{C}$, $\mathcal{D}$ of a P system, we say that $\mathcal{C}$ is a* restriction *of $\mathcal{D}$, in symbols $\mathcal{C} \sqsubseteq \mathcal{D}$, if $\mathcal{C}$ is obtained from $\mathcal{D}$ by deleting zero or more membranes, including their whole content (both objects and children membranes), and zero or more of the remaining objects.*

Being based on the subtree partial ordering of membrane structures and on the submultiset partial order, the relation $\sqsubseteq$ is also a partial order.

For the purposes of this paper, we consider as valid restricted configurations even those obtained by only keeping part of the environment and ignoring the membrane structure altogether. For instance, given the configuration

$$\mathcal{D} = \big[[a\ b]_k\ [c\ c]_\ell\ d\ d\ d\big]_h\ e\ f$$

the following are all valid restrictions of $\mathcal{D}$:

$$\mathcal{C}_1 = [[a\ b]_k\ d\ d]_h \qquad \mathcal{C}_2 = [[c\ c]_\ell]_h \qquad \mathcal{C}_3 = [d\ d\ d]_h\ e \qquad \mathcal{C}_4 = e\ f$$

A subclass of restricted configurations that we will focus on in this paper consists of the "small" configurations, which contain only one object, or up to two objects, if they are both located inside an internal membrane.

Definition 3. *We call a configuration $\mathcal{C}$ of an MSD P system Π a* small configuration *if it consists of the isolated object* yes *or* no *in the environment, or if it is of one of the forms $[a]_h$, $\big[[a]_k\big]_h$, or $\big[[a\ b]_k\big]_h$ where h is the outermost membrane of Π, k is the label of an internal membrane, and a, b are objects of the alphabet.*

A simple counting argument shows that the number of small configurations for an MSD P system is bounded by $(m^2 + m)\ell + 2 \in O(m^2\ell)$, where m is the size of the alphabet, and ℓ the number of labels, which corresponds to the initial number of membranes.

The following result shows that a halting computation of an MSD P system contains a sequence of small configurations which, alone, suffice to establish the result of the computation. This theorem is a stronger version of an analogous result for monodirectional P systems with charges [8, Lemma 1], where a *polynomial* number of objects and membranes were necessary and sufficient to decide the result of the computation.

Theorem 1. *Let Π be an accepting (resp., rejecting) MSD P system, and let $\mathcal{C}_1$ be a configuration reachable in any number of steps from the initial configuration $\mathcal{C}_0$ of Π. Let $\vec{\mathcal{C}} = (\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_t)$ be the halting subcomputation starting at $\mathcal{C}_1$. Then, there exists a sequence of small configurations $\vec{\mathcal{D}} = (\mathcal{D}_1, \ldots, \mathcal{D}_t)$ with $\mathcal{D}_i \sqsubseteq \mathcal{C}_i$ for $1 \le i \le t$ and $\mathcal{D}_t = \mathsf{yes}$ (resp., $\mathcal{D}_t = \mathsf{no}$) and a sequence of configurations $\vec{\mathcal{E}} = (\mathcal{E}_2, \ldots, \mathcal{E}_t)$ such that $\mathcal{D}_i \to \mathcal{E}_{i+1}$ and $\mathcal{D}_{i+1} \sqsubseteq \mathcal{E}_{i+1} \sqsubseteq \mathcal{C}_{i+1}$ for all $1 \le i < t$.*

Proof. We construct the sequences $\vec{\mathcal{D}} = (\mathcal{D}_1, \ldots, \mathcal{D}_t)$ and $\vec{\mathcal{E}} = (\mathcal{E}_2, \ldots, \mathcal{E}_t)$ by recursion on t. If $t = 1$ then $\mathcal{C}_1$ is already an accepting (resp., rejecting) configuration, and thus $\mathcal{D}_1 = \mathsf{yes} \sqsubseteq \mathcal{C}_1$ (resp, $\mathcal{D}_1 = \mathsf{no} \sqsubseteq \mathcal{C}_1$); in this case, the sequence $\vec{\mathcal{E}}$ is empty.

Now suppose $t > 1$. Then, the sub-computation $\vec{\mathcal{C}'} = (\mathcal{C}_2, \ldots, \mathcal{C}_t)$, i.e., the same computation as $\vec{\mathcal{C}}$ but starting from the second step, is a halting computation starting at a configuration reachable from the initial configuration $\mathcal{C}_0$ of Π. By induction hypothesis, there exists a sequence $\vec{\mathcal{D}'} = (\mathcal{D}_2, \ldots, \mathcal{D}_t)$ of small configurations and a sequence $\vec{\mathcal{E}'} = (\mathcal{E}_3, \ldots, \mathcal{E}_t)$ of configurations satisfying the statement of the theorem.

We construct $\mathcal{D}_1$ and $\mathcal{E}_2$ according to the form of $\mathcal{D}_2$ and the choice of rules that may have produced that configuration. The following list exhausts all possibilities.

(1) If $\mathcal{D}_2 = \mathsf{yes}$ (resp., $\mathcal{D}_2 = \mathsf{no}$), then there necessarily exists a send-out rule $[a]_h \to [\,]_h\ \mathsf{yes}$ (resp., $[a]_h \to [\,]_h\ \mathsf{no}$) which is applied in the step $\mathcal{C}_1 \to \mathcal{C}_2$. In this case, we let $\mathcal{D}_1 = [a]_h$ and $\mathcal{E}_2 = [\,]_h\ \mathsf{yes}$ (resp., $\mathcal{E}_2 = [\,]_h\ \mathsf{no}$).

(2) If $\mathcal{D}_2 = [a]_h$ and object a is produced by an object evolution rule $[b \to a\ w]_h$, then let $\mathcal{D}_1 = [b]_h$ and $\mathcal{E}_2 = [a\ w]_h$.

(3) If $\mathcal{D}_2 = [a]_h$ and object a is produced by a send-out rule $[b]_k \to [\,]_k\ a$, then let $\mathcal{D}_1 = \big[[b]_k\big]_h$ and $\mathcal{E}_2 = [[\,]_k\ a]_h$.

(4) If $\mathcal{D}_2 = [a]_h$ and object a is produced by a dissolution rule $[b]_k \to a$, then let $\mathcal{D}_1 = \big[[b]_k\big]_h$ and $\mathcal{E}_2 = [a]_h$.

(5) If $\mathcal{D}_2 = [a]_h$ and object a appeared inside an internal membrane k in $\mathcal{C}_1$, but fell out due to another object b applying a dissolution rule $[b]_k \to c$, then let $\mathcal{D}_1 = \big[[a\ b]_k\big]_h$ and $\mathcal{E}_2 = [a\ c]_h$.

(6) If $\mathcal{D}_2 = [a]_h$ and object a evolved from an object b appearing inside an internal membrane k in $\mathcal{C}_1$ using the rule $[b \to a\ w]_k$, and fell out due to another object c applying a dissolution rule $[c]_k \to d$, then let $\mathcal{D}_1 = \big[[b\ c]_k\big]_h$ and $\mathcal{E}_2 = [a\ w\ d]_h$.

(7) If $\mathcal{D}_2 = \big[[a]_k\big]_h$ and object a is produced by an evolution rule $[b \to a\ w]_k$, then let $\mathcal{D}_1 = \big[[b]_k\big]_h$ and $\mathcal{E}_2 = \big[[a\ w]_k\big]_h$.

(8) If $\mathcal{D}_2 = \big[[a]_k\big]_h$ and object a is produced by a division rule $[c]_k \to [a]_k\ [b]_k$, then let $\mathcal{D}_1 = \big[[c]_k\big]_h$ and $\mathcal{E}_2 = [[a]_k\ [b]_k]_h$.

(9) If $\mathcal{D}_2 = \big[[a\ b]_k\big]_h$ and objects a, b are produced by an object evolution rule $[c \to a\ b\ w]_k$, then let $\mathcal{D}_1 = \big[[c]_k\big]_h$ and $\mathcal{E}_2 = \big[[a\ b\ w]_k\big]_h$.

(10) If $\mathcal{D}_2 = \big[[a\ b]_k\big]_h$ and objects a, b are produced by two object evolution rules $[c \to a\ v]_k$ and $[d \to b\ w]_k$, then let $\mathcal{D}_1 = \big[[c\ d]_k\big]_h$ and $\mathcal{E}_2 = \big[[a\ v\ b\ w]_k\big]_h$.

(11) If $\mathcal{D}_2 = \big[[a\ b]_k\big]_h$, object a is produced by an object evolution rule $[c \to a\ w]_k$, and object b already appeared inside membrane k, then let $\mathcal{D}_1 = \big[[c\ b]_k\big]_h$ and $\mathcal{E}_2 = \big[[a\ w\ b]_k\big]_h$.

(12) If $\mathcal{D}_2 = \big[[a\ b]_k\big]_h$, object a is produced by a division rule $[c]_k \to [a]_k\ [d]_k$, and object b is produced by an object evolution rule $[e \to b\ w]_k$, then let $\mathcal{D}_1 = \big[[c\ e]_k\big]_h$ and $\mathcal{E}_2 = [[a\ b\ w]_k\ [d\ b\ w]_k]_h$.

$$\begin{tikzcd}
\mathcal{C}_0 \arrow[r] \arrow[d, squiggly] & \mathcal{C}_1 \arrow[r] \arrow[d, phantom, sloped, "\sqsupseteq"] & \mathcal{C}_2 \arrow[rr, dashed] \arrow[d, phantom, sloped, "\sqsupseteq"] & & \mathcal{C}_{t-1} \arrow[r] \arrow[d, phantom, sloped, "\sqsupseteq"] & \mathcal{C}_t \arrow[d, phantom, sloped, "\sqsupseteq"] \\
\mathcal{D}_0 \arrow[dr] & \mathcal{D}_1 \arrow[dr] & \mathcal{D}_2 \arrow[drr, dashed] & & \mathcal{D}_{t-1} \arrow[dr] & \mathcal{D}_t \\
& \mathcal{E}_1 \arrow[u, squiggly] & \mathcal{E}_2 \arrow[u, squiggly] & & \mathcal{E}_{t-1} \arrow[u, squiggly] & \mathcal{E}_t \arrow[u, squiggly]
\end{tikzcd}$$

Fig. 1. A computation $\vec{\mathcal{C}} = (\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{t-1}, \mathcal{C}_t)$ of Π and a sequence of configurations $\mathcal{C}_0 \rightsquigarrow \mathcal{D}_0 \to \mathcal{E}_1 \rightsquigarrow \mathcal{D}_1 \to \mathcal{E}_2 \rightsquigarrow \mathcal{D}_2 \to \cdots \to \mathcal{E}_{t-1} \rightsquigarrow \mathcal{D}_{t-1} \to \mathcal{E}_t \rightsquigarrow \mathcal{D}_t$ alternating restriction steps ($\rightsquigarrow$) and application of the rules from the corresponding steps of $\vec{\mathcal{C}}$ ($\to$) limited to the objects in $\mathcal{D}_i$. The diagram "commutes" in the sense that, starting from $\mathcal{C}_0$, the two paths either both lead to accepting configurations, or both to rejecting configurations. Notice that $\mathcal{E}_i \sqsubseteq \mathcal{C}_i$ for all i, since $\mathcal{E}_i$ is obtained from $\mathcal{D}_{i-1} \sqsubseteq \mathcal{C}_{i-1}$ and the P system is deterministic.

(13) If $\mathcal{D}_2 = \big[[a\ b]_k\big]_h$, object a is produced by a division rule $[c]_k \to [a]_k\ [d]_k$, and object b already appeared inside membrane k, then let $\mathcal{D}_1 = \big[[c\ b]_k\big]_h$ and $\mathcal{E}_2 = [[a\ b]_k\ [d\ b]_k]_h$.

It is easy to check by inspection that in all cases (1)–(13) we have $\mathcal{D}_1 \sqsubseteq \mathcal{C}_1$, $\mathcal{D}_1 \to \mathcal{E}_2$, and $\mathcal{D}_2 \sqsubseteq \mathcal{E}_2 \sqsubseteq \mathcal{C}_2$ as required. □

Fig. 1 shows the relationship between the computation $\vec{\mathcal{C}}$ and the sequences of configurations $\vec{\mathcal{D}}$ and $\vec{\mathcal{E}}$ in the statement of Theorem 1.

Notice that Theorem 1 does not directly provide an algorithm for computing *which* sequence $\vec{\mathcal{D}} = (\mathcal{D}_0, \ldots, \mathcal{D}_t)$ of small configurations gives the result of the computation. Furthermore, as will be shown in Section 4, it is not even sufficient to check that $\mathcal{D}_0 \sqsubseteq \mathcal{C}_0$ and $\mathcal{D}_t = \mathsf{yes}$ to guarantee the acceptance of Π, due to the possible interference from objects in $\vec{\mathcal{C}}$ not appearing in $\vec{\mathcal{D}}$.

Another fundamental limitation of MSD P systems is that they always halt in linear time with respect to the size of the alphabet, otherwise they would enter an infinite loop.

Theorem 2. *Every MSD P system halts within $2m$ steps, where m is the size of its alphabet.*

Proof. Assume that the P system is accepting, as the rejecting case is symmetrical. The object yes descends from an object a contained in the initial configuration of the P system via a sequence of rule applications. The object a is initially either inside the outermost membrane, or inside an elementary membrane immediately contained therein; in the latter case, either it reaches the outermost membrane via communication or by dissolving the membrane, or it reaches the outermost membrane due to *another* object dissolving the membrane.

If a is initially located inside the outermost membrane, then it can be rewritten a number of times by object evolution rules until an object b with an associated send-out rule $[b]_h \to [\,]_h$ yes is produced; notice that object evolution is the only way to produce b, as the outermost membrane cannot divide. In the shortest possible computation, the object b appears after a sequence of at most $m-1$ rewriting steps, where at least one new object is introduced at each step: indeed, since object evolution rules are context-free, each object either (i) becomes b within $m-1$ steps, or (ii) it stops evolving before $m-1$ steps without ever becoming b, or (iii) it enters an infinite rewriting loop. Case (iii) is impossible, since the P system is a recogniser; case (ii) is also impossible, since reaching b is necessary in order to send out the result yes. Hence case (i) must hold, and the computation accepts in at most m steps.

An analogous argument shows that a rule of the form $[a]_k \to [\,]_k\ b$ or $[a]_k \to b$ is applied inside the elementary membrane containing a after at most $m-1$ steps, if a is not initially located inside the outermost membrane. After further $m-1$ steps, the result is then sent out by a rule $[c]_h \to [\,]_h$ yes as detailed above. In this case, the accepting computation has a length at most $2m$. □

This result implies that the class of languages recognised by MSD P systems with no restriction on computation resources is the same as the class of languages they recognise in polynomial time.

Corollary 1. $\mathbf{DPMC}_{\mathcal{D}} = \mathbf{DMC}_{\mathcal{D}}$ *and* $\mathbf{DPMC}^{\star}_{\mathcal{D}} = \mathbf{DMC}^{\star}_{\mathcal{D}}$*, where* $\mathcal{D}$ *is the class of MSD P systems.* □

4 Dependency Graphs with Dissolution

Dependency graphs for P systems were introduced by Cordón-Franco et al. [3] as a way to track the objects in a P system without charges with non-cooperative rules and fixed membrane structures, and were later extended to membrane division rules [5], leaving out only membrane dissolution.

A dependency graph for a P system Π of this kind has, as vertices, all possible configurations of the form $\big[[\cdots\ [a]_{h_d}\ \cdots]_{h_1}\big]_{h_0}$, consisting of a linear sub-membrane structure of the membrane structure of Π, and a single object contained in the innermost membrane; a single object a in the environment (i.e., without any surrounding membrane) is also allowed[2]. Two configurations $\mathcal{V}, \mathcal{W}$ of the dependency graph are connected by an oriented edge if, when applying the rules of Π from configuration $\mathcal{V}$ (recall that, for a confluent P system, multiple choices are generally possible), we can reach a configuration $\mathcal{V}'$ containing $\mathcal{W}$, or $\mathcal{W} \sqsubseteq \mathcal{V}'$ in our notation (Definition 2).

[2] In the original notation [3,5] the surrounding membranes are left implicit, and thus the vertex $\big[[\cdots\ [a]_{h_d}\ \cdots]_{h_1}\big]_{h_0}$ is simply denoted by (a, h_d); an object a in the environment is denoted by (a, env).

The usefulness of the dependency graph for a P system Π without dissolution lies in the fact that it can be constructed in polynomial time from the description of Π, and that Π accepts if and only if there exists a vertex $\mathcal{V}$ contained in the initial configuration $\mathcal{C}_0$, in symbols $\mathcal{V} \sqsubseteq \mathcal{C}_0$, that is connected with a path to the node yes, representing the positive answer object in the environment[3]. This property can be easily checked in polynomial time, and this proves that the P conjecture is indeed true *limited to P systems without dissolution* [5].

The reason why checking reachability on dependency graphs suffices is due to the fact that such P systems lack *cooperation*, or, in other words, the only interaction between objects is due to the fact that the rules (excluding object evolution) can compete for the same membrane. Indeed, a theorem analogous to our Theorem 1, but stating that the result of the entire computation depends only on "very small configurations" containing *exactly one single object*, can be easily inferred from the above-mentioned result [5].

Allowing membrane dissolution breaks the entire reasoning, by introducing cooperation (or context-sensitiveness). Indeed, consider a configuration $\big[[a\ u]_k\big]_h$ with a dissolution rule $[a]_k \to b$ and zero or more object evolution rules rewriting the multiset u into v. This configuration leads to $[b\ v]_h$ in one step, and now the objects in v are subject to a potentially completely different set of rules, since the label of the membrane containing them was changed by the object a dissolving k. Thus, the objects in v have been affected by a, which is not possible in a P system without charges and without dissolution; furthermore, the influence of a on v cannot be represented by a standard dependency graph, as this tracks each object separately.

A way to generalise dependency graphs in order to take cooperation into account is to use larger configurations as vertices, containing multiple objects; as an extreme case, the whole set of configurations of Π could be kept as set of vertices. The side effect of this choice is the growth of the dependency graph. In general, a dependency graph keeping track of n objects per vertex has $\Theta(n^m)$ vertices, where m is the size of the alphabet; this value is exponential if $n \in \Theta(m)$. If there is no bound on the number of objects per vertex, the dependency graph might even become infinite.

In the case of MSD P systems, however, we can exploit Theorem 1 to keep the vertices of the dependency graph small, and thus reducing their number to polynomial, since at most two objects per vertex are needed even in the presence of dissolution.

Definition 4. *The* dependency graph $G(\Pi) = \big(V(\Pi), E(\Pi)\big)$ *for an MSD P system has as vertices* $V(\Pi)$ *all small configurations of* Π *and, as edges, the set* $E(\Pi) = \big\{(\mathcal{U}, \mathcal{V}) : \mathcal{U} \to \mathcal{C}$ *for some configuration* $\mathcal{C}$ *and* $\mathcal{V} \sqsubseteq \mathcal{C}\big\}$. *We denote by* $Y(\Pi)$ *the subset of* $V(\Pi)$ *consisting of all vertices connected with a path to the small configuration* yes.

Fig. 2 shows the dependency graph for a simple MSD P system using all available types of rule.

[3] That is, node $(\mathsf{yes}, \mathrm{env})$ in the original notation.

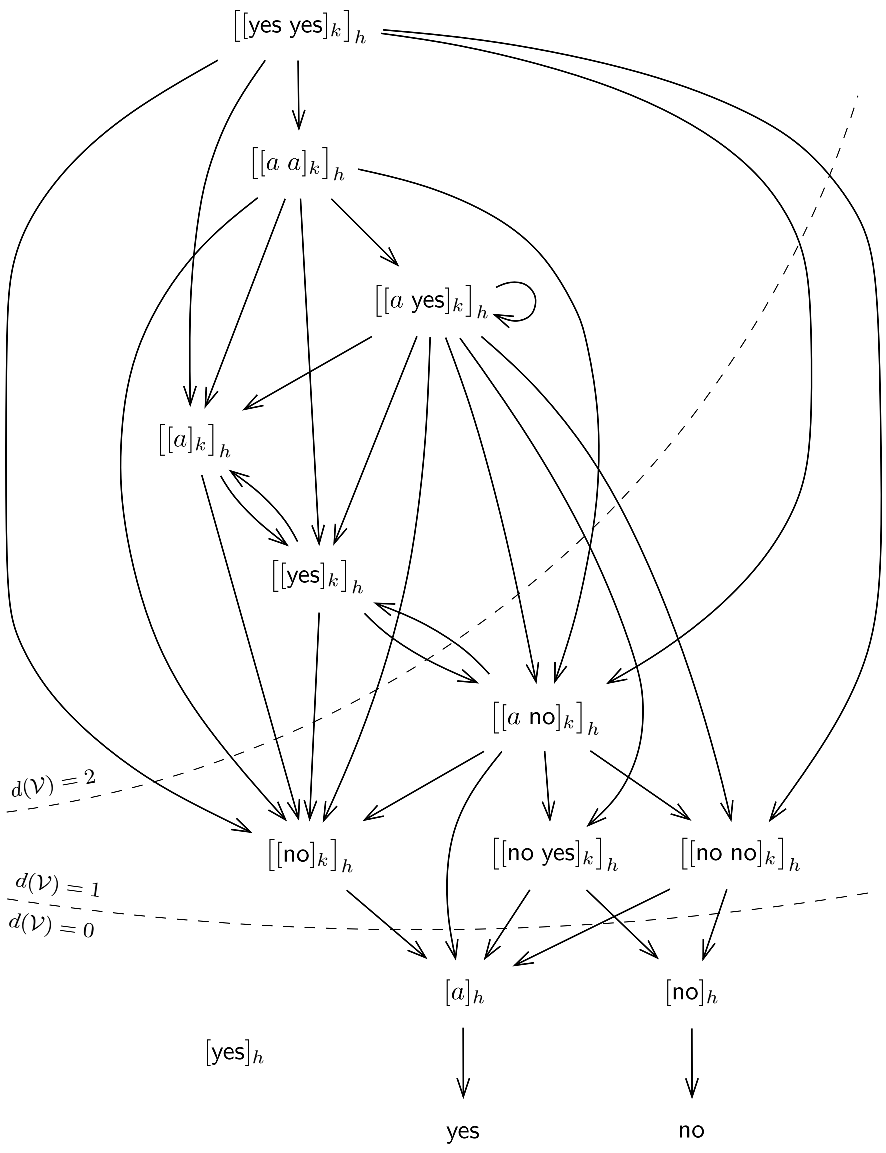

Fig. 2. Dependency graph for an MSD P system Π having alphabet $\Gamma = \{a, \mathsf{yes}, \mathsf{no}\}$ and rules $[a]_k \to [\mathsf{yes}]_k\ [\mathsf{no}]_k$, $[\mathsf{no}]_k \to a$, $[\mathsf{yes} \to a\ \mathsf{no}]_k$, $[\mathsf{no}]_h \to [\ ]_h\ \mathsf{no}$, $[a]_h \to [\ ]_h\ \mathsf{yes}$. Notice that this dependency graph is neither connected (the vertex $[\mathsf{yes}]_h$ is isolated), nor acyclic (it even includes a self-loop). Also notice that some vertices, such as $[[a\ \mathsf{no}]_k]_h$, show nondeterministic behaviour, since two rules can be applied; this does not necessarily contradict the determinism of Π, since these vertices $\mathcal{V}$ might not satisfy $\mathcal{V} \sqsubseteq \mathcal{C}$ for any configuration $\mathcal{C}$ reachable from the initial configuration. This P system accepts, for instance, from the initial configuration $\mathcal{C}_0 = [[\mathsf{no}]_k]_h$, but rejects from $\mathcal{C}_0 = [\mathsf{no}]_h$. The dashed lines are isolines grouping together the vertices by distance d (Definition 5).

Notice that $\mathsf{yes} \sqsubseteq \mathcal{C}$ if and only if the object yes is located in the environment of configuration $\mathcal{C}$, that is, if and only if $\mathcal{C}$ is an accepting configuration. Hence, the small configurations $\mathcal{V} \in Y(\Pi)$ are those that allow a configuration $\mathcal{C} \sqsupseteq \mathcal{V}$ to ultimately reach an accepting configuration, under certain conditions. Unfortunately, the mere inclusion $\mathcal{V} \sqsubseteq \mathcal{C}$, which allowed us to establish the result of P systems without dissolution, does not suffice in the case of MSD P systems, as shown by the following example.

Example 1. Consider an MSD P system Π with the rules

$$[a \to c]_k \qquad [b \to d]_k \qquad [c \to e]_k \qquad [d]_k \to f$$
$$[c]_h \to [\,]_h\ \mathsf{no} \qquad [e]_h \to [\,]_h\ \mathsf{yes} \qquad [g]_k \to f$$

The dependency graph for Π includes the following path:

$$\vec{\mathcal{D}} = \left(\big[[a\ b]_k\big]_h, \big[[c\ d]_k\big]_h, [e]_h, \mathsf{yes}\right)$$

If the initial configuration of Π is $\mathcal{C}_0 = \big[[a\ b]_k\big]_h$, then Π has indeed the accepting computation

$$\big[[a\ b]_k\big]_h \quad \to \quad \big[[c\ d]_k\big]_h \quad \to \quad [e\ f]_h \quad \to \quad [f]_h\ \mathsf{yes}$$

If, on the other hand, the initial configuration is $\mathcal{C}_0 = \big[[a\ b\ g]_k\big]_h$, i.e., also including the object g, the resulting computation is rejecting:

$$\big[[a\ b\ g]_k\big]_h \quad \to \quad [c\ d\ f]_h \quad \to \quad [d\ f]_h\ \mathsf{no}$$

This shows that it is not sufficient to find a vertex $\mathcal{V}$, in this case $\mathcal{V} = \big[[a\ b]_k\big]_h$, such that $\mathcal{V} \sqsubseteq \mathcal{C}_0$ and there exists a path from $\mathcal{V}$ to yes if dissolution rules are allowed, since objects such as d may interfere with the path leading to yes.

As a consequence, the use of small configurations alone does not suffice to correctly determine the result of P systems where dissolution is allowed. A property more complex than “being connected via a path to the vertex yes” needs to be checked for the vertices contained in the initial configuration $\mathcal{C}_0$. The rest of this section is devoted to establishing this property.

We refer to objects such as g in Example 1, which do *not* appear in a small configuration $\mathcal{V}$ connected by a path to the vertex yes, and yet interfere with that path, as *troublemakers for* $\mathcal{V}$. In order to give a formal definition of troublemakers, we first define a few related notions.

First of all, notice that when the current configuration of the P system contains a vertex $\mathcal{V}$ of the form $[a]_h$, with an object located inside the outermost membrane, then no troublemaker can exist, since the outermost membrane cannot be dissolved. This applies, of course, also when $\mathcal{V} = \mathsf{yes}$, that is, after having reached the halting configuration. For the remaining vertices, we define a notion of distance from such vertices without troublemakers.

Definition 5. *Let Π be an MSD P system; define the function $d\colon Y(\Pi) \to \mathbb{N}$ by letting $d(\mathcal{V}) = 0$ if $\mathcal{V} = \mathsf{yes}$ or $\mathcal{V} = [a]_h$ for some $a \in \Gamma$, and letting $d(\mathcal{V})$ be the distance (in terms of oriented edges) of $\mathcal{V}$ to the nearest small configuration of the form $[a]_h$ in the subgraph of $G(\Pi)$ induced by $Y(\Pi)$, otherwise.*

Fig. 2 also shows the distances of each node of the example dependency graph.

Among the troublemakers for a small configuration of the form $\mathcal{V} = \big[[a]_k\big]_h$ or $\mathcal{V} = \big[[a\ b]_k\big]_h$ there are the objects that dissolve membrane k, unless a or b themselves have dissolution rules. We refer to these objects with the symbol

$$\mathrm{dis}(k) = \{a \in \Gamma : \text{there exists a rule } [a]_k \to b \text{ for some } b \in \Gamma\}$$

Other troublemakers do not dissolve membrane k immediately, but rather produce the set of objects X that are troublemakers for vertices reachable from $\mathcal{V}$ with an edge. These may be produced either by object evolution rules, and we denote them by

$$\mathrm{evo}_k(X) = \{a \in \Gamma : \text{there exists a rule } [a \to b\ w]_k \text{ for some } b \in X\}$$

or by division rules, and we denote them by

$$\mathrm{div}_k(X) = \{a \in \Gamma : \text{there exists a rule } [a]_k \to [b]_k\ [c]_k \text{ for some } b, c \in X\}$$

Notice that send-out rules cannot produce troublemakers for any connected vertex, since the outermost membrane, where the result of the send-out appears, cannot be dissolved.

Now let $\mathcal{V}$ be a small configuration of the form $\big[[a]_k\big]_h$ or $\big[[a\ b]_k\big]_h$. In order to recursively compute its set of troublemakers, let us consider all vertices $\mathcal{W}$ with an edge $(\mathcal{V}, \mathcal{W}) \in E(\Pi)$. Among these, we keep only the vertices $\mathcal{W}_1, \ldots, \mathcal{W}_n$ having distance $d(\mathcal{W}_i) = d(\mathcal{V}) - 1$. The reason to exclude adjacent vertices $\mathcal{W}$ with $d(\mathcal{W}) \geq d(\mathcal{V})$ is that these require more computation steps to reach a troublemaker-free small configuration and, intuitively, the set of troublemakers of a vertex can increase with its distance. Furthermore, the troublemakers of $\mathcal{V}$ only depend on the *intersection* of the troublemakers of $\mathcal{W}_1, \ldots, \mathcal{W}_n$; indeed, a troublemaker for $\mathcal{W}_i$ that is not simultaneously a troublemaker for some $\mathcal{W}_j$ can be bypassed by following the edge $(\mathcal{V}, \mathcal{W}_j)$.

Based on this intuition we can now define, for each small configuration $\mathcal{V}$ on a path to yes, the notion of *set of troublemakers* $\mathrm{tm}(\mathcal{V})$ by recursion on the distance $d(\mathcal{V})$.

Definition 6. *Let Π be an MSD P system and let 2^Γ be the power set of its alphabet. Define the function $\mathrm{tm}\colon Y(\Pi) \to 2^\Gamma$ as*

$$\mathrm{tm}(\mathcal{V}) = \varnothing \tag{1}$$

if $d(\mathcal{V}) = 0$ or $d(\mathcal{V}) = 1$, and

$$\mathrm{tm}(\mathcal{V}) = \mathrm{dis}(k) \cup \mathrm{evo}_k\Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i)\Big) \cup \mathrm{div}_k\Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i)\Big) \tag{2}$$

if $d(\mathcal{V}) \geq 2$*, where* $\mathcal{W}_1, \ldots, \mathcal{W}_n$ *are all vertices in* $Y(\Pi)$ *such that* $(\mathcal{V}, \mathcal{W}_i) \in E(\Pi)$ *and* $d(\mathcal{W}_i) = d(\mathcal{V}) - 1$*.*

In fact, we can actually prove that the set of troublemakers $\mathrm{tm}(\mathcal{V})$ depends only on the distance $d(\mathcal{V})$ and on the label of the internal membrane, but not on the actual objects it contains.

Lemma 1. *Let* Π *be an MSD P system, let* $\mathcal{U}, \mathcal{V}$ *be two small configurations of* Π *such that, if both* $\mathcal{U}$ *and* $\mathcal{V}$ *contain two nested membranes, then the internal ones have the same label, and assume that* $d(\mathcal{U}) = d(\mathcal{V})$*. Then* $\mathrm{tm}(\mathcal{U}) = \mathrm{tm}(\mathcal{V})$*.*

Proof. By induction on $d(\mathcal{U})$. If $d(\mathcal{U}) = d(\mathcal{V}) = 0$ or $d(\mathcal{U}) = d(\mathcal{V}) = 1$, then $\mathrm{tm}(\mathcal{U}) = \varnothing = \mathrm{tm}(\mathcal{V})$. If $d(\mathcal{U}) = d(\mathcal{V}) \geq 2$ then, according to equation (2), we have

$$\mathrm{tm}(\mathcal{U}) = \mathrm{dis}(k) \cup \mathrm{evo}_k \Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \Big) \cup \mathrm{div}_k \Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \Big)$$
$$\mathrm{tm}(\mathcal{V}) = \mathrm{dis}(k) \cup \mathrm{evo}_k \Big(\bigcap_{i=1}^{m} \mathrm{tm}(\mathcal{Z}_i) \Big) \cup \mathrm{div}_k \Big(\bigcap_{i=1}^{m} \mathrm{tm}(\mathcal{Z}_i) \Big)$$

with $d(\mathcal{W}_1) = \cdots = d(\mathcal{W}_n) = d(\mathcal{Z}_1) = \cdots = d(\mathcal{Z}_m) < d(\mathcal{U})$. Hence, by induction hypothesis we have $\mathrm{tm}(\mathcal{W}_1) = \cdots = \mathrm{tm}(\mathcal{W}_n) = \mathrm{tm}(\mathcal{Z}_1) = \cdots = \mathrm{tm}(\mathcal{Z}_m)$, which implies $\mathrm{tm}(\mathcal{U}) = \mathrm{tm}(\mathcal{V})$. $\square$

The following monotonicity result formalises the intuitive notion that, with the increase of $d(\mathcal{V})$, the possibilities for external objects to interfere with a path on the dependency graph may also increase.

Lemma 2. *Let* Π *be an MSD P system, let* $\mathcal{U}, \mathcal{V}$ *be two small configurations of* Π *such that, if both* $\mathcal{U}$ *and* $\mathcal{V}$ *contain two nested membranes, the internal ones have the same label, and assume that* $d(\mathcal{U}) \leq d(\mathcal{V})$*. Then* $\mathrm{tm}(\mathcal{U}) \subseteq \mathrm{tm}(\mathcal{V})$*.*

Proof. It suffices to show that $\mathrm{tm}(\mathcal{U}) \subseteq \mathrm{tm}(\mathcal{V})$ when $d(\mathcal{U}) = d(\mathcal{V}) - 1$. We prove it by induction on $d(\mathcal{U})$. If $d(\mathcal{U}) = 0$ or $d(\mathcal{U}) = 1$, then $\mathrm{tm}(\mathcal{U}) = \varnothing$, which is always a subset of $\mathrm{tm}(\mathcal{V})$.

If $d(\mathcal{U}) \geq 2$ then $d(\mathcal{V}) > 2$, and both $\mathrm{tm}(\mathcal{U})$ and $\mathrm{tm}(\mathcal{V})$ are computed using equation (2):

$$\mathrm{tm}(\mathcal{U}) = \mathrm{dis}(k) \cup \mathrm{evo}_k \Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \Big) \cup \mathrm{div}_k \Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \Big)$$
$$\mathrm{tm}(\mathcal{V}) = \mathrm{dis}(k) \cup \mathrm{evo}_k \Big(\bigcap_{i=1}^{m} \mathrm{tm}(\mathcal{Z}_i) \Big) \cup \mathrm{div}_k \Big(\bigcap_{i=1}^{m} \mathrm{tm}(\mathcal{Z}_i) \Big)$$

Since $d(\mathcal{Z}_1) = \cdots = d(\mathcal{Z}_m) = d(\mathcal{V}) - 1 = d(\mathcal{U})$, then by Lemma 1 we have $\mathrm{tm}(\mathcal{Z}_1) = \cdots = \mathrm{tm}(\mathcal{Z}_m) = \mathrm{tm}(\mathcal{U})$, and thus

$$\begin{aligned}\mathrm{tm}(\mathcal{V}) &= \mathrm{dis}(k) \cup \mathrm{evo}_k \Big(\bigcap_{i=1}^{m} \mathrm{tm}(\mathcal{U}) \Big) \cup \mathrm{div}_k \Big(\bigcap_{i=1}^{m} \mathrm{tm}(\mathcal{U}) \Big) \\ &= \mathrm{dis}(k) \cup \mathrm{evo}_k \big(\mathrm{tm}(\mathcal{U}) \big) \cup \mathrm{div}_k \big(\mathrm{tm}(\mathcal{U}) \big)\end{aligned}$$

Since $d(\mathcal{W}_1), \dots, d(\mathcal{W}_n) = d(\mathcal{U}) - 1$, we have $\mathrm{tm}(\mathcal{W}_1), \dots, \mathrm{tm}(\mathcal{W}_n) \subseteq \mathrm{tm}(\mathcal{U})$ by induction hypothesis, and thus $\bigcap_{i=1}^n \mathrm{tm}(\mathcal{W}_i) \subseteq \mathrm{tm}(\mathcal{U})$, which implies

$$\mathrm{tm}(\mathcal{V}) \supseteq \mathrm{dis}(k) \cup \mathrm{evo}_k \Big(\bigcap_{i=1}^n \mathrm{tm}(\mathcal{W}_i) \Big) \cup \mathrm{div}_k \Big(\bigcap_{i=1}^n \mathrm{tm}(\mathcal{W}_i) \Big) = \mathrm{tm}(\mathcal{U})$$

as needed. □

We call a configuration $\mathcal{C}$ of an MSD P system Π *untroubled* if it contains a vertex $\mathcal{V} \in V(\Pi)$ that is connected via a path to the vertex yes, and none of the troublemakers of $\mathcal{V}$. Notice, however, that a vertex can be contained multiple times in a given configuration, and each occurrence may or may not be subject to interference by its troublemakers. For instance, let $\mathcal{V} = \big[[a\ b]_k\big]_h$ with $\mathrm{tm}(\mathcal{V}) = \{c\}$ and $\mathcal{C} = \big[[a\ a\ b]_k\ [a\ b\ c]_k\big]_h$; the small configuration $\mathcal{V}$ occurs three times in $\mathcal{C}$ (twice in the left membrane k, and once in the right one), but only one occurrence may have interference, since the troublemaker c does not occur in the left membrane k. In order to formalise the notion of untroubled membrane, given a configuration of an MSD P system

$$\mathcal{C} = \big[[w_1]_{k_1}\ \cdots\ [w_n]_{k_n}\ w_0\big]_h\ w$$

we say that $\mathcal{L} \sqsubseteq \mathcal{C}$ is a *linear restriction of* $\mathcal{C}$ if either $\mathcal{L} = w$, or $\mathcal{L} = [w_0]_h$, or $\mathcal{L} = \big[[w_i]_{k_i}\big]_h$ for some $1 \le i \le n$. In a linear restriction $\mathcal{L}$, only one membrane contains objects, and thus the occurrences of a small configuration $\mathcal{V} \sqsubseteq \mathcal{L}$ are either all subject to interference from a troublemaker, or none of them is.

Definition 7. *We say that a configuration $\mathcal{C}$ of an MSD P system is* untroubled *if there exists a linear restriction $\mathcal{L} \sqsubseteq \mathcal{C}$ and a small configuration $\mathcal{V} \in Y(\Pi)$ such that $\mathcal{V} \sqsubseteq \mathcal{L}$ and no object $a \in \mathrm{tm}(\mathcal{V})$ appears in $\mathcal{L}$.*

The following results show that the property of being untroubled propagates forwards along a computation step and thus, recursively, all the way to the halting configuration.

Lemma 3. *Let Π be an MSD P system, let $\mathcal{C}$ be an untroubled configuration reachable from the initial configuration $\mathcal{C}_0$, and let $\mathcal{C} \to \mathcal{D}$. Then $\mathcal{D}$ is also untroubled.*

Proof. Since $\mathcal{C}$ is untroubled, there exists a linear restriction $\mathcal{L} \sqsubseteq \mathcal{C}$ and a small configuration $\mathcal{V} \sqsubseteq \mathcal{L}$ such that no object of $\mathrm{tm}(\mathcal{V})$ belongs to $\mathcal{L}$.

If $d(\mathcal{V}) \le 2$, then there exists a small configuration $\mathcal{W}$ with $d(\mathcal{W}) \le 1$ such that $\mathcal{V} \to \mathcal{V}' \sqsupseteq \mathcal{W}$ for some configuration $\mathcal{V}'$, otherwise $d(\mathcal{V})$ would be at least 3. But then $\mathrm{tm}(\mathcal{W}) = \varnothing$ and thus $\mathcal{D}$ is untroubled.

If $d(\mathcal{V}) \ge 3$, then $\mathcal{V} \to \mathcal{V}' \sqsupseteq \mathcal{W}$ for some $\mathcal{W}$ such that $d(\mathcal{W}) = d(\mathcal{V}) - 1 \ge 2$, and there exists a linear restriction $\mathcal{M} \sqsubseteq \mathcal{D}$ with $\mathcal{W} \sqsubseteq \mathcal{M}$. By contradiction suppose that, for all such $\mathcal{W}$, there exists $a \in \mathrm{tm}(\mathcal{W})$ with a belonging to $\mathcal{M}$.

This object a itself cannot appear in $\mathcal{L}$: since we have $\mathrm{tm}(\mathcal{W}) \subseteq \mathrm{tm}(\mathcal{V})$ by Lemma 2, we would have simultaneously $a \in \mathrm{tm}(\mathcal{V})$ and a in $\mathcal{L}$, contradicting our assumptions on $\mathcal{V}$. Hence, the object a must be generated by an object evolution or division rule triggered by an object b in $\mathcal{L}$.

If it is generated by an object evolution rule, then $b \in \mathrm{evo}_k\big(\mathrm{tm}(\mathcal{W})\big)$; since tm only depends on the distance d (Lemma 1), this means that we have $b \in \mathrm{evo}_k\big(\mathrm{tm}\big(\bigcap_{i=1}^n \mathcal{W}_i\big)\big)$, where the $\mathcal{W}_i$ are all small configurations, including $\mathcal{W}$, such that $(\mathcal{V}, \mathcal{W}_i) \in E(\Pi)$ with $d(\mathcal{W}_i) = d(\mathcal{V}) - 1$. But then $b \in \mathrm{tm}(\mathcal{V})$ while simultaneously appearing in $\mathcal{L}$, once again contradicting the hypotheses on $\mathcal{V}$.

Suppose, instead, that the object a is generated by a division rule of the form $[b]_h \to [a]_h\ [c]_h$. We necessarily have $c \in \mathrm{tm}(\mathcal{W})$, because otherwise there would exist a linear restriction $\mathcal{M}' \sqsubseteq \mathcal{D}$ distinct from $\mathcal{M}$ with $\mathcal{M}'$ containing the other copy of $\mathcal{W}$ resulting from the division, but without any object in $\mathrm{tm}(\mathcal{W})$. But then we have both $a, c \in \mathrm{tm}(\mathcal{W})$, which implies that

$$b \in \mathrm{div}_k\big(\mathrm{tm}(\mathcal{W})\big) = \mathrm{div}_k\Big(\bigcap_{i=1}^n \mathrm{tm}(\mathcal{W}_i)\Big) \subseteq \mathrm{tm}(\mathcal{V})$$

once again contradicting the hypotheses on $\mathcal{V}$.

This shows that no object $a \in \mathrm{tm}(\mathcal{W})$ can belong to $\mathcal{M}$, and thus $\mathcal{D}$ is untroubled. $\square$

Lemma 4. *Let Π be an MSD P system with untroubled initial configuration $\mathcal{C}_0$. Then Π accepts.*

Proof. Suppose that Π has the halting computation $\vec{\mathcal{C}} = (\mathcal{C}_0, \ldots, \mathcal{C}_t)$ with untroubled $\mathcal{C}_0$. By Lemma 3, all other configurations of $\vec{\mathcal{C}}$, and in particular $\mathcal{C}_t$, are also untroubled. This means that there exist a linear restriction $\mathcal{L} \sqsubseteq \mathcal{C}_t$ and a small configuration $\mathcal{V} \in Y(\Pi)$ such that $\mathcal{V} \sqsubseteq \mathcal{L}$ and no $a \in \mathrm{tm}(\mathcal{V})$ appears in $\mathcal{L}$. The only small configuration in $Y(\Pi)$ where no rule is applicable is yes. This proves that the P system accepts. $\square$

The property of being untroubled does not only propagate forwards, but also *backwards*, all the way to the initial configuration.

Lemma 5. *Let Π be an MSD P system, let $\mathcal{C}$ be a configuration reachable from the initial configuration $\mathcal{C}_0$, and let $\mathcal{C} \to \mathcal{D}$ with $\mathcal{D}$ untroubled. Then $\mathcal{C}$ is also untroubled.*

Proof. Since $\mathcal{D}$ is untroubled, there exist a linear restriction $\mathcal{M} \sqsubseteq \mathcal{D}$ and a small configuration $\mathcal{W} \sqsubseteq \mathcal{M}$ such that no object $a \in \mathrm{tm}(\mathcal{W})$ appears in $\mathcal{M}$.

Since $\mathcal{W}$ contains at most two objects, it is generated by at most two rules applied in $\mathcal{C}$; this means that there exist a linear restriction $\mathcal{L} \sqsubseteq \mathcal{C}$ and a small configuration $\mathcal{V} \sqsubseteq \mathcal{L}$ such that $\mathcal{V} \to \mathcal{V}' \sqsupseteq \mathcal{W}$ for some configuration $\mathcal{V}'$. Let us consider the set $\mathrm{tm}(\mathcal{V})$. If $d(\mathcal{V}) \leq 1$, this set is computed according to equation (1), and then $\mathrm{tm}(\mathcal{V}) = \varnothing$ and $\mathcal{C}$ is untroubled.

If $d(\mathcal{V}) \geq 2$, the set $\mathrm{tm}(\mathcal{V})$ is computed according to equation (2):

$$\mathrm{tm}(\mathcal{V}) = \mathrm{dis}(k) \cup \mathrm{evo}_k \Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \Big) \cup \mathrm{div}_k \Big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \Big)$$

In this case we have $\mathcal{W} = \big[[a]_k\big]_h$ or $\mathcal{W} = \big[[a\ b]_k\big]_h$. Let us consider an object $a \in \mathrm{tm}(\mathcal{V})$. If $a \in \mathrm{dis}(k)$, then a does not appear in $\mathcal{L}$, because membrane k survived the transition $\mathcal{V} \to \mathcal{V}' \sqsupseteq \mathcal{W}$. If $a \in \mathrm{evo}_k \big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \big)$ appeared in $\mathcal{L}$, then it would evolve into an object of $\mathcal{M}$ belonging to $\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \subseteq \mathrm{tm}(\mathcal{W})$, which contradicts the hypotheses on $\mathcal{W}$ and is thus impossible. Similarly, if $a \in \mathrm{div}_k \big(\bigcap_{i=1}^{n} \mathrm{tm}(\mathcal{W}_i) \big)$ appeared in $\mathcal{L}$, then $\mathcal{M}$ would contain one of the two objects on the right-hand side of the division rule involving a, and that object would belong to $\mathrm{tm}(\mathcal{W})$; this contradicts the hypotheses on $\mathcal{W}$, and thus is also impossible. Hence, no object in $\mathrm{tm}(\mathcal{V})$ appears in $\mathcal{L}$, and thus $\mathcal{C}$ is untroubled. □

Lemma 6. *Let Π be an MSD P system with troubled initial configuration $\mathcal{C}_0$. Then Π rejects.*

Proof. Suppose that Π has the accepting computation $\vec{\mathcal{C}} = (\mathcal{C}_0, \dots, \mathcal{C}_t)$. Then $\mathsf{yes} \sqsubseteq \mathcal{C}_t$ and $\mathrm{tm}(\mathsf{yes}) = \varnothing$, which means that $\mathcal{C}_t$ is untroubled. But we have $\mathcal{C}_0 \to \mathcal{C}_1 \to \cdots \to \mathcal{C}_t$, and by Lemma 5 all previous configurations, including $\mathcal{C}_0$, are untroubled, which contradicts our hypotheses. □

Hence, deciding whether an MSD P system accepts coincides with checking the troublemakers in its initial configuration.

Theorem 3. *An MSD P system accepts if and only if its initial configuration is untroubled.* □

If membrane dissolution rules are disallowed, the term $\mathrm{dis}(k)$ is always empty in equation (2), and thus $\mathrm{tm}(\mathcal{V}) = \varnothing$ for all small configurations $\mathcal{V}$; this implies that all configurations are untroubled (Definition 7). Furthermore, the small configurations of the form $\big[[a\ b]_k\big]_h$ can be ignored in favour of the smaller configurations $\big[[a]_k\big]_h$ and $\big[[b]_k\big]_h$, since a configuration with two objects is only required when the dissolution caused by one allows the other object to eventually evolve into yes (cases (5) and (6) of Theorem 1). This shows that, in the absence of dissolution, the acceptance condition of Theorem 3 corresponds to the existence of a path from a small configuration of the form $[a]_h$ or $\big[[a]_k\big]_h$, included in the initial configuration of the P system, to the small configuration yes. This is exactly the original acceptance condition for standard dependency graphs [3,5].

An analysis of the computational resources required in order to check the condition of Theorem 3 finally allows us to show that MSD P systems characterise the complexity class $\mathbf{P}$.

Theorem 4. $\mathbf{DMC}_{\mathcal{D}}^{[\star]} = \mathbf{DPMC}_{\mathcal{D}}^{[\star]} = \mathbf{P}$, *where $\mathcal{D}$ is the class of MSD P systems and $[\star]$ denotes optional semi-uniformity.*

Proof. Since the following inclusions hold

$$\begin{tikzcd}[column sep=small]
 & \mathbf{DPMC}^\star_\mathcal{D} \arrow[dr, phantom, sloped, "\subseteq"] & \\
\mathbf{P} \subseteq \mathbf{DPMC}_\mathcal{D} \arrow[ur, phantom, sloped, "\subseteq"] \arrow[dr, phantom, sloped, "\subseteq"] & & \mathbf{DMC}^\star_\mathcal{D} \\
 & \mathbf{DMC}_\mathcal{D} \arrow[ur, phantom, sloped, "\subseteq"] &
\end{tikzcd}$$

it suffices to prove $\mathbf{DMC}^\star_\mathcal{D} \subseteq \mathbf{P}$. Let $\boldsymbol{\Pi}$ be a semi-uniform family of recogniser MSD P systems constructed in polynomial time by Turing machine H.

Given an input string $x \in \Sigma^\star$, simulate H on x in polynomial time in order to obtain the description of the P system Π_x.

The alphabet of Π_x and its set of rules have polynomial size, and thus the dependency graph $G(\Pi)$ can be constructed in polynomial time (see Gutiérrez-Naranjo et al. [5] for more details).

The set of vertices $Y(\Pi)$ connected to the vertex yes, and thus the subgraph of $G(\Pi)$ induced by them, can be computed in polynomial time by exploring the transposed dependency graph (i.e., with all edges reversed) starting from the vertex yes.

The value $d(\mathcal{V})$ can then be computed in polynomial time for each $\mathcal{V} \in Y(\Pi)$ by using an all-pairs shortest path algorithm on the subgraph induced by $Y(\Pi)$, and then choosing the distance from the closest vertex of the form $[a]_h$ or yes.

The troublemakers $\mathrm{tm}(\mathcal{V})$ are computed recursively in polynomial time using equations (1) and (2) on the subgraph induced by $Y(\Pi)$ and based on the distance function d.

Finally, for each of the (polynomially many) small configurations $\mathcal{V}$ and all (polynomially many) linear restrictions $\mathcal{L} \sqsubseteq \mathcal{C}_0$, where $\mathcal{C}_0$ is the initial configuration of Π, we can check whether $\mathcal{V} \sqsubseteq \mathcal{L}$ and simultaneously no object $a \in \mathrm{tm}(\mathcal{V})$ appears in $\mathcal{L}$. If this happens at least once, the input string x is accepted, and otherwise rejected. □

5 Conclusions and Open Problems

We have proved that families of monodirectional, deterministic, shallow P systems with active membranes without charges (MSD P systems) characterise the complexity class **P**, both with a polynomial-time uniformity and a polynomial-time semi-uniformity condition. This solves an open special case of the P conjecture and shows that dissolution does not always allow us to break the **P** barrier, even if object evolution, send-out, and membrane division rules are allowed.

In order to prove this result, we developed a generalisation of an important membrane computing tool, already exploited for several **P** upper bound results for P systems without charges: namely, dependency graphs. By proving that a constant number of objects govern the result of the computation, we were able to construct polynomial-size dependency graphs that include dissolving membranes. By checking a property of dependency graphs more complex than the original one, but still verifiable in polynomial time, we can establish the result of the computation of an MSD P system without resorting to expensive full simulations.

Even if we only focused on P systems of depth at most one in this paper, our approach should be straightforward to generalise to MSD P systems of any constant depth, as long as membrane division is only applicable to the initial elementary membranes, by tracking a constant number of objects larger than two. If elementary division rules are associated to membranes that only become elementary during the computation, there is the additional problem of establishing when this is the case for the membranes we are tracking (since "being elementary" depends on untracked membranes having dissolved). Here, we believe it is worth investigating a combination of our approach with the one proposed by Woods et al. [14], which deals correctly with dissolution and elementary division in P systems of arbitrary depth in the absence of other types of rules.

We further conjecture that it is possible to replace the determinism constraint with the standard confluence condition. This seems, however, to make the forwards and backwards propagation of the property of being untroubled harder to prove, due to the possibility of multiple computations.

We also remarked that several other generalisations of dependency graphs, for use with other variants of P systems, are possible. Although it is unclear whether this approach is powerful enough to solve the remaining open cases of the P conjecture, an interesting open problem is establishing which classes of P systems admit polynomial-size dependency graphs with polynomial-time computable accepting conditions. A related question is whether it is possible to find classes of P systems with larger-than-polynomial dependency graphs that do not require a complete exploration, and thus still allow their result to be established in polynomial time.

References

1. Alhazov, A., Freund, R.: On the efficiency of P systems with active membranes and two polarizations. In: Mauri, G., Păun, Gh., Pérez-Jiménez, M.J., Rozenberg, G., Salomaa, A. (eds.) Membrane Computing, 5th International Workshop, WMC 2004. Lecture Notes in Computer Science, vol. 3365, pp. 146–160. Springer (2005)
2. Alhazov, A., Pérez-Jiménez, M.J.: Uniform solution to QSAT using polarizationless active membranes. In: Durand-Lose, J., Margenstern, M. (eds.) Machines, Computations, and Universality, 5th International Conference, MCU 2007, Lecture Notes in Computer Science, vol. 4664, pp. 122–133. Springer (2007)
3. Cordón-Franco, A., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Exploring computation trees associated with P systems. In: Mauri, G., Păun, Gh., Pérez-Jiménez, M.J., Rozenberg, G., Salomaa, A. (eds.) Membrane Computing, 5th International Workshop, WMC 2004. Lecture Notes in Computer Science, vol. 3365, pp. 278–286. Springer (2005)
4. Gazdag, Z., Kolonits, G.: Remarks on the computational power of some restricted variants of P systems with active membranes. In: Leporati, A., Rozenberg, G., Salomaa, A., Zandron, C. (eds.) Membrane Computing, 17th International Conference, CMC 2016. Lecture Notes in Computer Science, vol. 10105, pp. 209–232. Springer (2017)

5. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A., Romero-Campero, F.J.: P systems with active membranes, without polarizations and without dissolution: A characterization of P. In: Calude, C.S., Dinneen, M.J., Păun, Gh., Pérez-Jímenez, M.J., Rozenberg, G. (eds.) Unconventional Computation, 4th International Conference, UC 2005. Lecture Notes in Computer Science, vol. 3699, pp. 105–116. Springer (2005)
6. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A., Romero-Campero, F.J.: On the power of dissolution in P systems with active membranes. In: Freund, R., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) Membrane Computing, 6th International Workshop, WMC 2005. Lecture Notes in Computer Science, vol. 3850, pp. 224–240. Springer (2006)
7. Leporati, A., Manzoni, L., Mauri, G., Porreca, A.E., Zandron, C.: Simulating elementary active membranes, with an application to the P conjecture. In: Gheorghe, M., Rozenberg, G., Sosík, P., Zandron, C. (eds.) Membrane Computing, 15th International Conference, CMC 2014, Lecture Notes in Computer Science, vol. 8961, pp. 284–299. Springer (2014)
8. Leporati, A., Manzoni, L., Mauri, G., Porreca, A.E., Zandron, C.: Monodirectional P systems. Natural Computing 15(4), 551–564 (2016)
9. Murphy, N., Woods, D.: Active membrane systems without charges and using only symmetric elementary division characterise P. In: Eleftherakis, G., Kefalas, P., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) Membrane Computing, 8th International Workshop, WMC 2007. Lecture Notes in Computer Science, vol. 4860, pp. 367–384 (2007)
10. Murphy, N., Woods, D.: The computational power of membrane systems under tight uniformity conditions. Natural Computing 10(1), 613–632 (2011)
11. Păun, Gh.: Further twenty six open problems in membrane computing. In: Gutíerrez-Naranjo, M.A., Riscos-Nuñez, A., Romero-Campero, F.J., Sburlan, D. (eds.) Proceedings of the Third Brainstorming Week on Membrane Computing. pp. 249–262. Fénix Editora (2005)
12. Pérez-Jiménez, M.J., Romero-Jiménez, A., Sancho-Caparrini, F.: Complexity classes in models of cellular computing with membranes. Natural Computing 2(3), 265–284 (2003)
13. Sosík, P., Rodríguez-Patón, A.: Membrane computing and complexity theory: A characterization of PSPACE. Journal of Computer and System Sciences 73(1), 137–152 (2007)
14. Woods, D., Murphy, N., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Membrane dissolution and division in P. In: Calude, C.S., Félix Costa, J., Dershowitz, N., Freire, E., Rozenberg, G. (eds.) Unconventional Computation, 8th International Conference, UC 2009, Lecture Notes in Computer Science, vol. 5715. Springer (2009)